Numerical simulation of viscoelastic freesurface flows using a streamfunction/log conformation formulation and the volumeoffluid method

Comminal, Raphael Benjamin

Publication date:
2015

Document Version
Publisher's PDF, also known as Version of record

Numerical simulation of viscoelastic free-surface flows using a streamfunction/log-conformation formulation and the volume-of-fluid method

Raphaël Comminal

Ph.D. thesis

Technical University of Denmark
Department of Mechanical Engineering

May, 2015

Preface

The work presented in this thesis was carried out at the Department of Mechanical Engineering (MEK) at the Technical University of Denmark (DTU), in the period 2011–2015. This thesis is submitted in partial fulfillment of the requirement for the Doctor of Philosophy degree.

The work was funded by the Danish Agency for Science Technology and Innovation (FTP) (Contract No. 09-072888), which is part of the Danish Council for Independent Research (DFF). This Ph.D. project was a part of a larger project called “Optimized Processing of Multimaterial Architectures for Functional Ceramics” (OPTIMAC). The project consisted of three work packages and DTU-MEK was involved in WP1 whose focus was on process modelling. This Ph.D. project was carried out in close collaboration between DTU-MEK and the Department of Energy Conversion and storage (DTU-Energy). This work was supervised by Professor Jesper H. Hattel (DTU-MEK) and co-supervised by Assistant Professor Jon Spangenberg (DTU-MEK) and Professor Nini Pryds (DTU-Energy). This work was also financially supported by DTU-MEK.

I would like to express my deepest gratitude to Professor Jesper H. Hattel for his guidance and enthusiasm about my work. His highly inspiring and motivating talks have helped me a lot throughout my entire studies.

I would also like to thank Professor Nini Pryds together with the other members of the OPTIMAC project, in particular Dr. Masoud Jabbari, Dr. Dhavanesan K. Ramachandran and Dr. Andreas Kaiser for their interest and critical discussion on my work, as well as Dr. Frank Clemens who has welcomed me at his laboratory at the Swiss Federal Laboratories for Materials Science and Technology (EMPA) in Zurich.

I would like to express my special gratitude to Assistant Professor Jon Spangenberg for his involvement in my Ph.D. project. His help and encouragement has contributed a lot to the quality of my work and the writing of this thesis.

I would like to thank Dr. Jesper Thorborg for helping out with any kind of support, as well as all the colleagues, current and former Ph.D. students and Post-Doc researchers of the Process Modelling group of Professor Jesper H. Hattel for creating a nice working environment.

Finally, I wish to express my deepest appreciation to my family and my friends for having supported me during these 4 years.

Raphaël Comminal
Kongens Lyngby
May 2015

Abstract

This thesis presents a new numerical algorithm for the simulation of two-dimensional multiphase viscoelastic flows. The simulation of viscoelastic flows has both a scientific importance and practical implications in polymer processing. This work has put the emphasis on the extrusion of polymeric materials, where viscoelastic effects cause dynamical instabilities, despite the very simple geometry. This thesis reviews the popular differential constitutive models derived from molecular theories of dilute polymer solutions, polymer networks, and entangled polymer melts, as well as the inelastic phenomenological models describing shear–thinning and viscoplastic (yield stress) fluids, based on the generalized Newtonian fluid model. In addition, the numerical issues related to the high Weissenberg number problem, and its remedy with the log–conformation representation, are discussed.

The proposed algorithm utilizes a new streamfunction/log–conformation scheme. The drawbacks of the classical velocity–pressure decoupled method, which is by far the most popular approach, are remedied with the pure streamfunction formulation, which is derived from the pressureless vorticity-based methods. The implicit pure streamfunction formulation is formally more accurate than the velocity–pressure decoupled method, because it is immune of decoupling errors. Moreover, the absence of decoupling enhances the stability of the calculation. The governing equations (conservation laws and constitutive models) are discretized with the finite–volume method, on a Cartesian grid. Discrete curl operators are applied to the discretized momentum equations in order to obtain the matrix system of the discrete streamfunction variables. The coupling of the streamfunction/log–conformation scheme with adaptive under-relaxation and adaptive time-stepping yield a robust and efficient viscoelastic flow solver algorithm. The potential extension of the method to three-dimensional simulations is also discussed in this thesis.

Bi-phasic/free-surface flows are modelled with the Volume–of–Fluid (VOF) method, and the standard piecewise–linear–interface–construction technique. In addition, a new Cellwise Conservative Unsplit (CCU) advection scheme is presented. The CCU scheme updates the liquid volume fractions based on cellwise backward-tracking of the liquid volumes. The algorithm calculates non-overlapping and conforming adjacent donating regions, which ensures the boundedness and conservativeness of the liquid volume. As a result, the CCU advection scheme is overall more accurate in classical benchmark tests, than the other state-of-the-art multi-dimensional VOF–advection schemes. In complex flows, the convergence rate of the CCU scheme with mesh refinements is between 2 and 3. Moreover, the remaining geometrical errors are mostly due to the inability of the standard piecewise linear interface to represent subgrid material topologies (i.e. high curvatures and thin material filaments), rather than the proposed CCU advection scheme.

This thesis reports examples of numerical simulations of the Oldroyd–B liquid, calculated with the proposed streamfunction/log–conformation/VOF–CCU methodology, implemented in Matlab. A thorough investigation of the viscoelastic flow in the lid-driven cavity is conducted. The streamfunction/log–conformation shows second-order accuracy and numerical stability at very large time-step increments, which demonstrates the robustness of

the scheme. The numerical results at moderate Weissenberg numbers are in good agreement with the literature. Moreover, the enhancement of numerical stability, with the streamfunction/log–conformation scheme, makes it possible to simulate elastic instabilities at high Weissenberg numbers. Quasi-periodic elastic instabilities at the upstream corner appear to be a mechanism that dissipates the stored elastic energy. The simulations of viscoelastic flows in the planar 4:1 contraction are also in good agreement with data in the literature. Finally, preliminary simulations of extrudate swelling show that the fracture melt extrusion defect could be caused by instabilities in the stress layer at the surface of the die, triggered at moderate Weissenberg numbers.

Resumé

Denne afhandling præsenterer en ny numerisk algoritme der kan anvendes til a simulere hvordan multifase viskoelastiske materialer flyder i to dimensioner. Simuleringer af viskolelastiske materialers flydeevne har både en videnskabelig relevans og en praktisk betydning for processer der involvere polymere. Den viskoelastiske opførelse giver anledning til dynamisk ustabilitet specielt i ekstrudering. De populære konstitutive modeller for fluider bestående af polymere er opsummeret. De numeriske problemstillinger der relaterer sig til det høje Weissenberg nummer problem er diskuteret og løsningen er fundet ved hjælp af log-tilpasningsformuleringen. Den foreslåede numeriske algoritme anvender også den trykløse strømfunktion formulering til at finde strømningsprofilerne. Denne formulering er mere stabil og præcis end de klassiske metoder der adskiller tryk- og hastighedsberegningen. Den nye strømfunktion/log-tilpasningsalgoritme giver sammen med en selvjusterende dæmpningsfaktor og tidskridt en robust og effektiv viskoelastisk løser. To faser/den frie overflade er modelleret med Volumen-Af-Fluid metoden og en ny cellevis konservativ ikke splittet advektions rutine, der bevarer væskens volumen andel og er mere præcis end andre nye procedurer. Simuleringerne af viskoelastiske sammenligningstests er i god overensstemmelse med tidligere udgivet resultater. Forbedringen af den numeriske stabilitet vha. strømfunktionen/log-tilpasningsalgoritmen gør det muligt at simulere elastisk ustabilitet. Dette er den fysiske mekanisme der opløser den lagrede elastiske energi.

Table of contents

1 Introduction **1**

1.1 Rheology and polymer processing 1

1.1.1 Capillary extrusion 2

1.1.2 Co-extrusion 4

1.2 State of the art in computational rheology 6

1.2.1 Lagrangian methods 6

1.2.2 Eulerian methods 8

1.2.3 Discrete particle methods 13

1.3 Current and future challenges 13

1.4 Aim and structure of the thesis 14

1.5 References 16

2 Constitutive modeling **23**

2.1 Generalized Newtonian fluid models 23

2.1.1 Power-law fluids 24

2.1.2 Viscoplastic fluids 25

2.2 Viscoelastic fluid models 26

2.2.1 Differential viscoelastic models 27

2.2.2 The high Weissenberg number problem 34

2.2.3 The log–conformation representation 35

2.3 References 36

3 Conservation laws **39**

3.1 Velocity–pressure formulation 39

3.1.1 Pseudo–compressibility methods 40

3.1.2 Velocity–pressure segregation methods 41

3.1.3 Drawbacks of the velocity–pressure formulation in viscoelastic flows 43

3.2 Vorticity formulation 45

3.3 Two-dimensional pure streamfunction formulation 46

3.4 References 47

4 Discretization methods **51**

4.1 Discrete curl operators 51

4.2 Finite-volume discretization 54

4.3 Solution algorithm . . . 58
4.4 Adaptive time-stepping . . . 59
4.5 Potential extension to three-dimensions . . . 61
4.6 References . . . 63

5 Multiphase modeling **65**
5.1 Eulerian surface-tracking techniques . . . 65
5.2 Geometrical interface reconstruction . . . 66
5.3 Advection schemes . . . 69
5.3.1 Directional-split advection scheme . . . 69
5.3.2 Cellwise conservative unsplit advection scheme . . . 71
5.3.2.1 Backward-tracing . . . 71
5.3.2.2 Geometric volume flux correction . . . 75
5.3.2.3 Cellwise VOF update . . . 78
5.3.3 Numerical tests . . . 78
5.3.3.1 Pre-image calculation test . . . 78
5.3.3.2 Single vortex deformation test . . . 81
5.4 Rule of mixture . . . 91
5.5 References . . . 94

6 Numerical results **99**
6.1 Single-phase viscoelastic flows in a lid-driven cavity . . . 99
6.1.1 Orders of convergence . . . 101
6.1.2 Simulations at moderate Weissenberg numbers . . . 104
6.1.3 Simulations at high Weissenberg numbers . . . 112
6.2 Single phase viscoelastic flows through a 4:1 planar contraction . . . 118
6.3 Simulation of the extrudate swelling of an Oldroyd–B liquid . . . 127
6.4 References . . . 135

7 Conclusions **139**
7.1 Summary . . . 139
7.2 Perspectives . . . 140

Appendices **143**

A Paper–I **145**

B Paper–II **153**

C Paper–III **167**

D Paper–IV **197**

Chapter 1

Introduction

This chapter presents an introduction to the thesis: Numerical simulation of viscoelastic free-surface flows using a streamfunction/log-conformation formulation and the volume-of-fluid method. The simulations of multiphase viscoelastic flows have a wide range of applications in natural science and engineering. This work has put the emphasis on the simulation of polymeric liquids, and their applications in Polymer Processing. An introduction to the importance of Rheology in Polymer Processing is given in section 1.1. The state of the art in Computational Rheology is reviewed in section 1.2. Afterwards, section 1.3 provides the author's opinion on the current and future challenges that faces the simulations of viscoelastic flows. Finally, section 1.4 presents the specific objectives of this work, and gives an overview of the structure of the thesis.

1.1. Rheology and polymer processing

Rheology is the science of the deformation and flow of matter. It is at the interface between Mechanics and Material Science, since it deals with the study of the constitutive behaviors of complex materials, and the influences of their microstructures. It can be approached both as an experimental and a theoretical discipline, which aims at developing realistic models describing complex materials. Computational Rheology concerns with the development of accurate and efficient calculation algorithms, using advanced mathematical and computational techniques, for the prediction of complex constitutive behaviors. Viscoelastic fluid dynamics, in particular, which exhibits different flow features than for Newtonian fluids, is an important subset of Computational Rheology (Crochet & Walters 1993).

Typical examples of viscoelastic liquids are polymer solutions, molten plastics, multigrade oils, contact adhesives, synthetic resins, printing inks, body fluids, etc. Amorphous materials in their glass–liquid phase transition (within the interval or the glass temperature and the melting temperature) also exhibit viscoelastic behaviors, since they are neither completely solids nor completely liquids. Potential fields of applications for the simulations of viscoelastic flows include polymer processing, bio-fluidic, micro-fluidic, chemical engineering, food industry, and much more.

Phenomenologically, viscoelastic materials have stress responses that do not only depend on their current deformations, but also on their deformation history. Thus, time is a key parameter; and viscoelastic materials have different stress responses depending on the time-scales of the deformation. They typically exhibit stress relaxation, creep deformation and shape memory. In contrast to Newtonian fluids, elastic liquids build-up *normal stress differences* when deformed (in both shear and shearfree flows). The normal stresses produce various

effects, such as rod climbing (also known as the Weissenberg effect), extrudate swell, elastic recoil, tubeless siphon, etc. (Bird, Armstrong & Hassager 1987). The elastic forces also generate secondary flows and *elastic instabilities* (Larson 1992; Shaqfeh 1996; Denn 2004) in various geometries. At large Reynolds numbers, viscoelastic flow are subject to *turbulence drag reduction* (Graham 2004). Moreover, the possibility of a turbulent flow regime due to elastic forces at low Reynolds number, the *elastic turbulence*, has recently been discovered (Groisman & Steinberg 2000; Morozov & van Saarloos 2007). Furthermore, the effects of viscoelasticity are magnified in the micro-fluidics (Li et al. 2012). Thus, the development of accurate calculation algorithms is important from both fundamental and practical points of view.

The rheological properties of polymeric materials are crucial parameters for their processability (i.e. the flowability). Moreover, the viscoelastic properties are responsible for complex flow features such as flow instabilities. The extensional rheological properties are fundamental in the manufacturing processes involving stretching of the materials, as for instance *fiber spinning*, *film casting*, *blow molding*, etc. (Jung & Hyun 2006). The draw resonance and the three-dimensional helicoidal instabilities are cause by viscoelasticity (Demay & Agassant 2014). Moreover, the shear rheological behavior plays a key role in *injection molding* and *calendaring* manufacturing processes. In injection molding for instance, the rheological characterization of the polymer melt (which is temperature-dependent) is essential in order to design the molding process. The mold-filling pattern can be simulated with the Computer-Aided Engineering softwares that include appropriated rheological models, in order to detect potential molding defects due to improper filling and freezing. Moreover, the coupling of the flow solver with a thermal solver and a stress solver can give predictions of fiber orientation, residual stress, shrinkage and warpage of the molded part, birefringence properties of the final product, etc. Hence, Computational Rheology is a useful tool for the optimization of the mold design and the processing parameters, to improve productivity and to reduce energy composition of the machine. Inelastic flow solvers—assuming purely viscous fluids—are widely available. However, taking viscoelastic effects into account is still non-conventional, although it gives more accurate predictions.

The *extrusion* through capillary dies is an interesting process which has been of special focus in this work, because it gives insight into the complex flow phenomena of polymeric fluids, in a simple geometry. The next subsections review various flow instabilities occurring in extrusion, due to the rheology (i.e. shear-thinning, viscoelastic, etc.), which limit the processing rate. In addition, multi-layered *co-extrusion* provides an understanding of interfacial instabilities in stratified parallel flows.

1.1.1. Capillary extrusion

The productivity of many continuous manufacturing processes for plastic material is limited by flow instabilities, which happen at a high processing speed, i.e. for a large apparent shear rate at the wall of the die. Among all the plastic manufacturing processes, extrusion is one of the processes which have been the most investigated. Two distinct instability phenomena (or extrusion defects) with different mechanisms (which can also be combined) are documented in the literature. The first phenomenon, the *sharkskin instability*, is a surface defect occurring at moderate extrusion speeds. It results in an irregular surface of the extrudate, with scratches and superficial cracks; see Figure 1.1. There is a consensus that the sharkskin occurs at the die exit, where the material near the wall is pulled out. Thus, the sharkskin depends on the extensional rheological properties. The second phenomenon, the *fracture melt instability*, occurs at larger extrusion speeds than the sharkskin. At the onset of instability, the shape of the extrudate starts to undulate with a regular wave length that is twice the capillary diameter (Bertola et al. 2003), see Figure 1.2. The amplitude of the oscillations increases with the extrusion speed and at some point the fluid is extruded in the shape of a regular helix, which gradually evolves into a chaotic regime, at larger flow rates. Piau, El Kissi and Tremblay (1990) have observed a correlation between the undulations of the extrudate and upstream flow instabilities inside the reservoir, which they

measured by particle image velocimetry (PIV). Their measurements show that the fracture melt is caused by asymmetry flows in the reservoir, where regular oscillations produce a helix extrudate, and a chaotic flow regime lead to an irregular extrusion. Identical conclusions were reached from the PIV measurements of the reservoir flow instabilities by Nigen et al. (2003). These flow instabilities in the reservoir are similar to the elastic instabilities in the 4:1 contraction flow, simulated by Afonso et al. (2011). In addition, the weakly nonlinear stability analysis of Meulenbroek et al. (2003) shows that a bulk instability similar to the fracture melt could be produced by an intrinsic subcritical instability of the viscoelastic Poiseuille flows (independently of the reservoir), see also the essay of Morozov and van Saarloos (2007) about elastic turbulence in parallel flows.

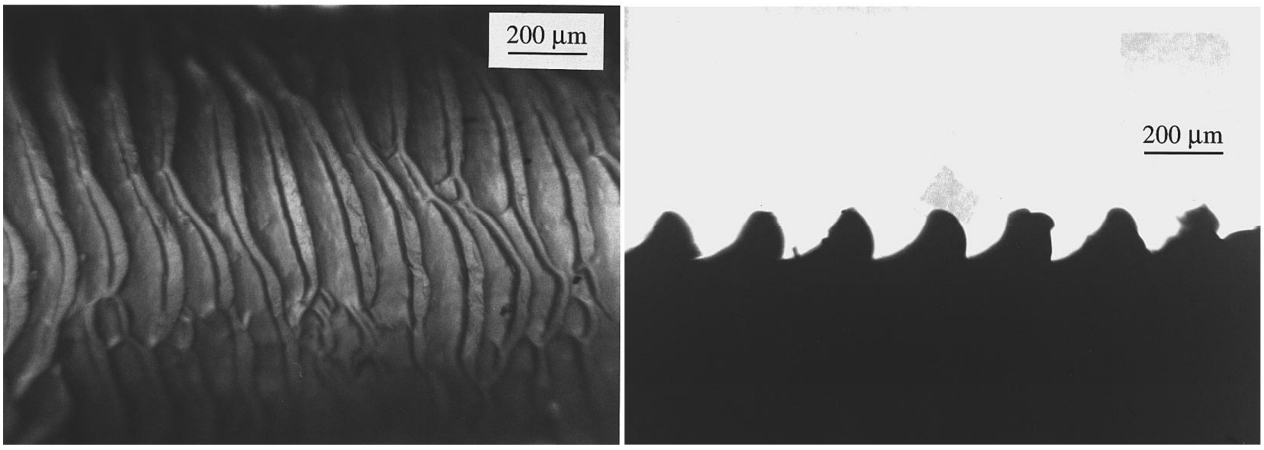


Figure 1.1: Sharkskin surface defect during extrusion of an LLDPE resin flowing from left to right: (left) view of the surface, (right) view of the longitudinal cut (Wang, Drda & Inn 1996).

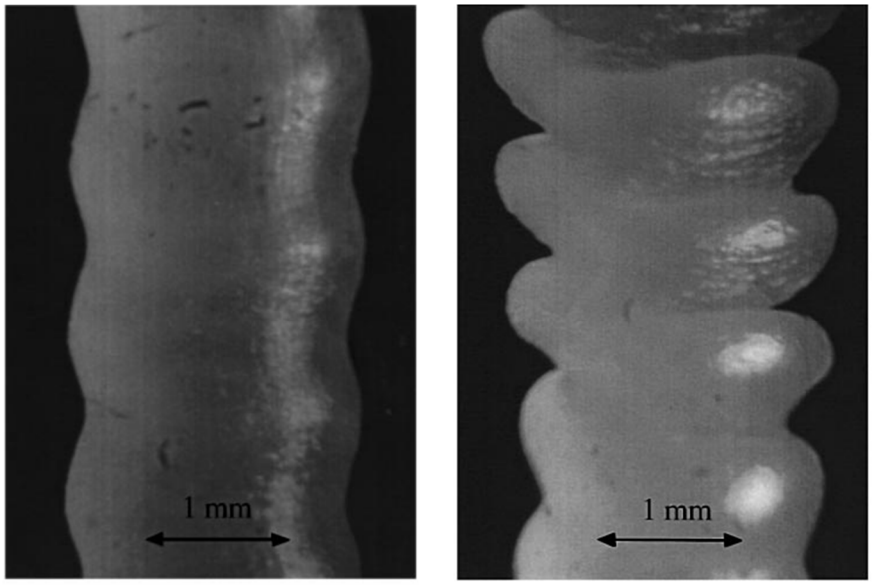


Figure 1.2: Fracture melt instability during extrusion of HDPE at 185°C through an axisymmetric capillary with 1mm in diameter and 5mm in length, for different apparent shear rates at the wall (Piau, Nigen & El Kissi 2000).

Wall slip is another phenomenon playing an important role in the flow stability, since it determines the boundary conditions of the flows; see the review of Denn (2001) about the wall slip effects in extrusion. The more recent review of Hatzikiriakos (2012) about the wall slip of molten polymers draws a complex picture of the polymer slip. Two mechanisms of slip are identified: the weak slip occurs because of flow-induced *desorption* (chain detachment) at the wall, and the strong slip is due to the *disentanglement* in the polymer chains in the bulk from a monolayer of absorbed chains, see Figure 1.3. However, it is difficult to measure experimentally the slip velocity and to isolate the two mechanisms. Indirect measurements reveal more complex constructive relations than the idealized Navier slip law, with an exponential dependency of the slip velocity on the wall shear stress. Moreover, experimental data shows that the transition from a dominant mechanism to another can produce stick–slip instabilities (Hatzikiriakos & Dealy 1992). Additionally, the numerical analysis of Georgiou have demonstrated that non-monotonous slip laws can lead to self-sustained oscillations, causing unstable flow regimes in the extrusion of compressible inelastic fluids (Georgiou & Crochet 1994; Georgiou 2003; Taliadorou, Georgiou & Alexandrou 2007) and incompressible viscoelastic fluids (Georgiou 1996; Brasseur et al. 1998). Moreover, such non-monotonous slip laws have been derived from a molecular model by Yarin and Graham (1998). A complete numerical model of the capillary flow, including nonlinear slip, viscoelasticity and the compressibility effect in the reservoir, has been presented by Shore at al. (1997). Their model predicts an oscillating regime due to the stick-slip behavior, as well as a regime of bulk flow instabilities because of viscoelasticity/compressibility interactions.

1.1.2. Co-extrusion

Co-extrusion is a common manufacturing process to produce parts with multilayer architectures and different functional materials. During co-extrusion, the different materials are injected simultaneously inside the same extrusion die; see for instance the microfabrication of Hoy et al. (1998) with ceramics compounds. The stability of the multilayer flow depends on the rheological properties of the extruded fluids.

Differences in the viscosity or the elastic properties of the fluids can produce secondary flow in the cross-section of the extrusion (Dooley & Rudolph 2003). A viscosity stratification in the non-axisymmetric profile is responsible for the *viscous encapsulation* phenomenon, where the less viscous material (i.e. with higher fluidity) wraps around the more viscous one (Han 1973; Southern & Ballman 1975), see Figure 1.4. The apparent viscosities of the fluids at the interface depend on the shear-rate, since most polymeric liquids exhibit a power-law shear thinning behavior (i.e. their apparent shear viscosities decrease with an increasing shear rate). Moreover, the different materials generally have different power indices, thus, the apparent viscosity ratio of the two material dependents on the flow rate and the thickness of each layer. Therefore, stable co-extrusion requires a fine tailoring of the material compounds and tuning of the extrusion rates (Ismael et al. 2009).

The stratification of the elastic properties (e.g. elastic moduli, relaxation times) of the co-extruded fluids can also produce deformations of the interface, called *elastic rearrangement*, due to an elastic stress in the cross-section of the extrudate (Anderson, Dooley & Meijer 2006; Yue et al. 2008), see Figure 1.5.

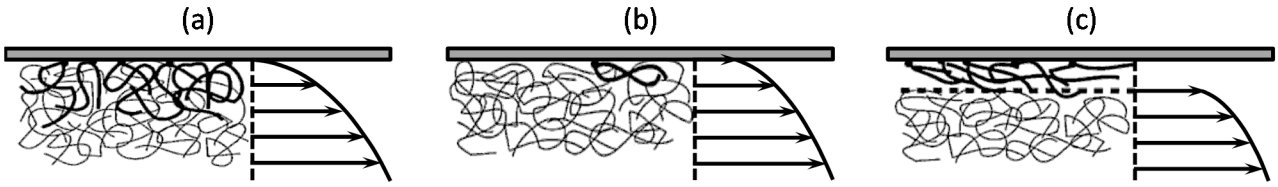
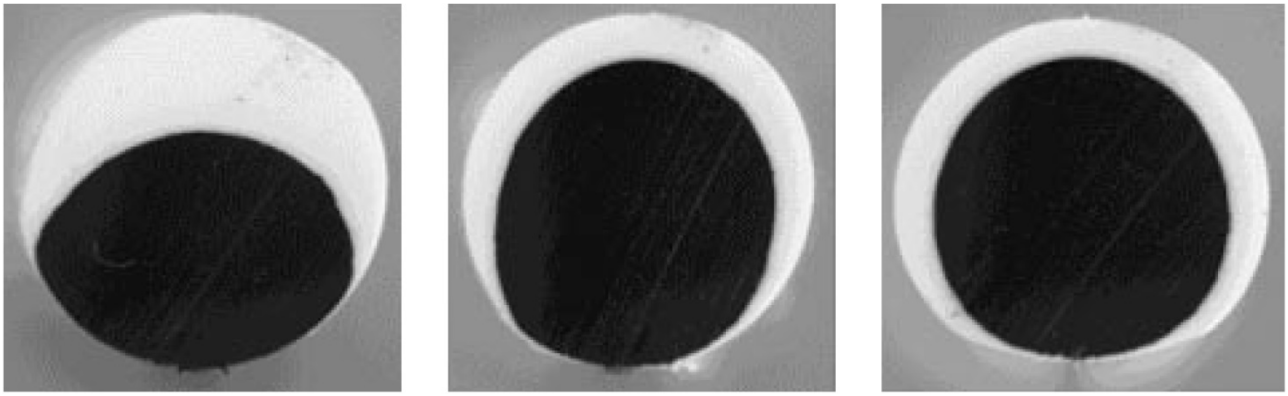
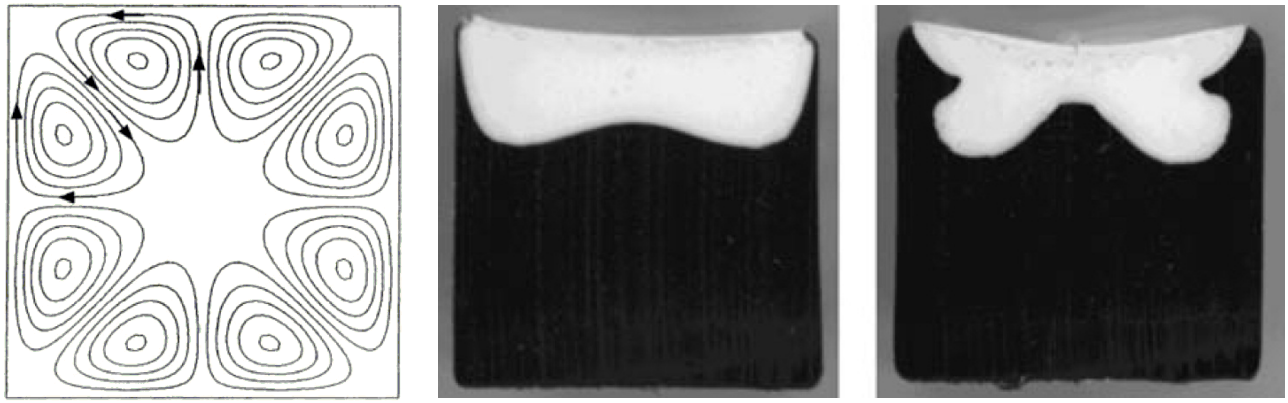


Figure 1.3: (a) Entangled polymer chain with no slip, (b) Weak slip due to chain desorption, (c) Strong slip due to the disentanglement of the bulk from absorbed chains at the wall.



(a) (b) (c)

Figure 1.4: Viscous encapsulation in bilayer co-extrusion of PE resins: (a) cross-section at the die entry; (b) cross-section at the middle of the die; (c) cross-section at the die exit (Dooley & Rudolph 2003).



(a) (b) (c)

Figure 1.5: Elastic rearrangement in bilayer co-extrusion of PE resins: (a) theoretical secondary flow produced by elastic forces; (b) cross-section at the die entry; (c) cross-section at the die exit (Dooley & Rudolph 2003).

In addition to cross-sectional deformations, the co-extruded profile can also be subject to *interfacial instabilities* in the longitudinal direction. The stability of the Couette and Poiseuille co-planar flows of different fluids to short-wave and long-wave perturbations has been studied analytically with linear and non-linear stability theories (Yih 1967; Joseph, Renardy & Renardy 1984; Yiantsios & Higgins 1988; Chen 1991; Hinch, Harris & Rallison 1992; Chen & Zhang 1993; Pinarbasi & Liakopoulos 1995). Several factors affect the co-flow stability: the viscosity ratio, the flow rates, the shear-thinning properties, the viscoelastic properties, the yield stresses, etc. The stability analysis provides stability diagrams that show the stable and unstable regions of the flow, for different sets of parameters.

Extensive experimental investigations on the interfacial stability of multi-layer polymer co-extrusion have been conducted by Wilson and Khomami (1992; 1993; Khomami & Wilson 1995). Their experiment consisted in monitoring the evolutions of controlled sinusoidal disturbances introduced in the flow rates of the layers. Their experimental results verified the founding of the linear stability theories, and showed that the non-linear supercritical instabilities evolve with shapes that resemble the Kelvin–Helmholtz instabilities, see Figure 1.6.

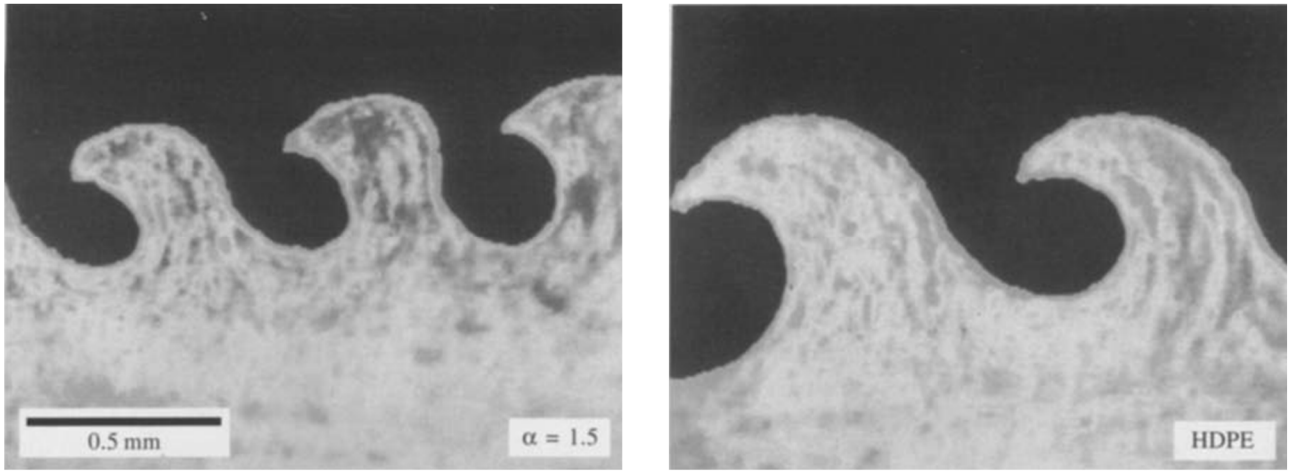


Figure 1.6: Developement of the longitudinal supercritical interfacial instability caused by small simusoidal disturbances of PP/HDPE bilayer flows from left to right (Khomami & Ranjbaran 1997).

1.2. State of the art in computational rheology

Computational Rheology deals with the constitutive modeling of all the materials with complex behaviors, ranging from purely elastic solids to purely Newtonian fluids. Consequently, different approaches have been used in computational rheology. Most of them belong either to the family of the *Lagrangian methods* or the family of the *Eulerian methods*. The Lagrangian methods are generally used in Computational Solid Mechanics (CSM), while the Eulerian methods are the preferred approach in Computational Fluid Dynamics (CFD).

Finally, an alternative to the Lagrangian and Eulerian methods is the use of *discrete particle methods*, where the motion of the material is not tracked by the use of a mesh, but with discrete particles. The discrete particle methods have the ability to model easily multiphase flows. However, the meshless approach remains non-conventional in Computational Rheology. The following subsections briefly review the use of these three approaches to the modelling of viscoelastic flows with interfaces or free surfaces.

1.2.1 Lagrangian methods

In the Lagrangian methods, the mesh follows the deformations of the material (similarly as in computational solid mechanics). With these techniques, the edges of the mesh coincide with the free-surfaces or the interfaces between different materials, for instance see Figure 1.7. Therefore, the Lagrangian methods naturally track free-surfaces and interfaces, without any additional calculation. The Lagrangian methods are typically combined with the finite–element method; however, it is also possible to solve the flow with finite–volume solvers. There are several methods to update the mesh, e.g. the conformal mapping technique, the arbitrary-Lagrangian-Eulerian method and the adaptive remeshing techniques.

The Lagrangian *conformal mapping technique* displaces the vertices of the mesh according to the local displacement of the material. This numerical technique was successfully used by Crochet and Keunings (1980; 1982) to simulate the extrudate swelling of viscoelastic materials; see for instance Figure 1.8. However, in the case of large deformations, the conformal mapping technique may produce highly distorded meshes. In general, distorded elements with large aspect ratio, warpage or skewness produce large numerical errors. In addition, the finite–element solver might be prone to spurious deformations, due to element locking or hourglass modes, if not handled properly.

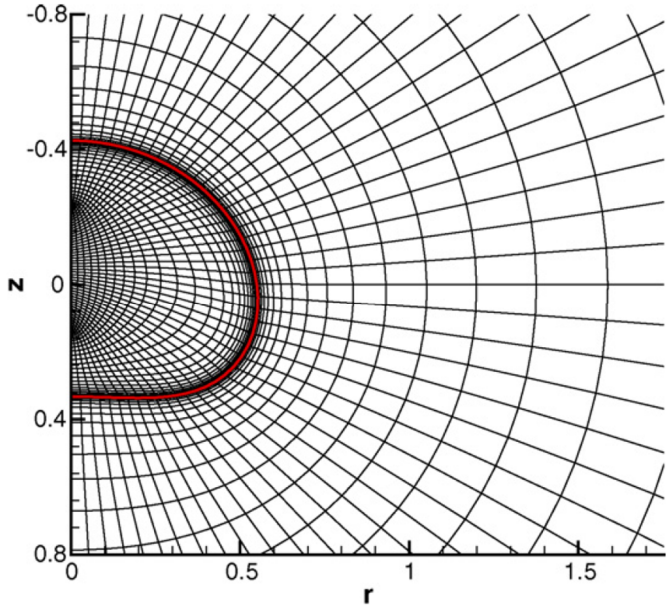


Figure 1.7: Simulation of the motion of a viscoelastic drop with a Lagrangian method using the conformal mapping technique (You, Borhan & Haj-Hariri 2008).

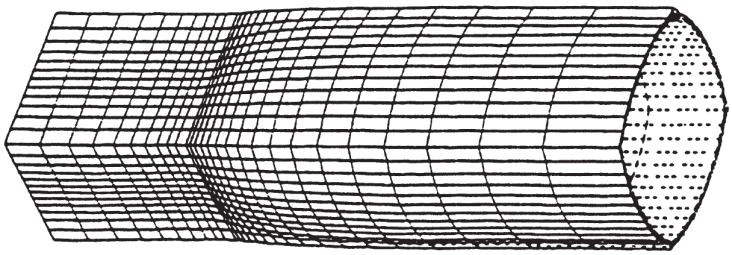


Figure 1.8: Three-dimensional simulation of the extrudate swell of a viscoelastic material exiting a rectangular die (Normandin et al. 1999).

In the *arbitrary-Lagrangian-Eulerian method*, only the mesh vertices at the interface or at the free boundaries follow the local displacement of the material, while the mesh displacement inside the bulk material is independent on the material motion (Étienne, Hinch & Li 2006). An optimum mesh displacement is calculated in order to reduce the mesh distortion. The arbitrary-Lagrangian-Eulerian method extends the possibility of application of Lagrangian methods, as compared to conformal mapping techniques. However, the optimization of the mesh displacement requires the solving of an additional system of equations, which increases the computational effort.

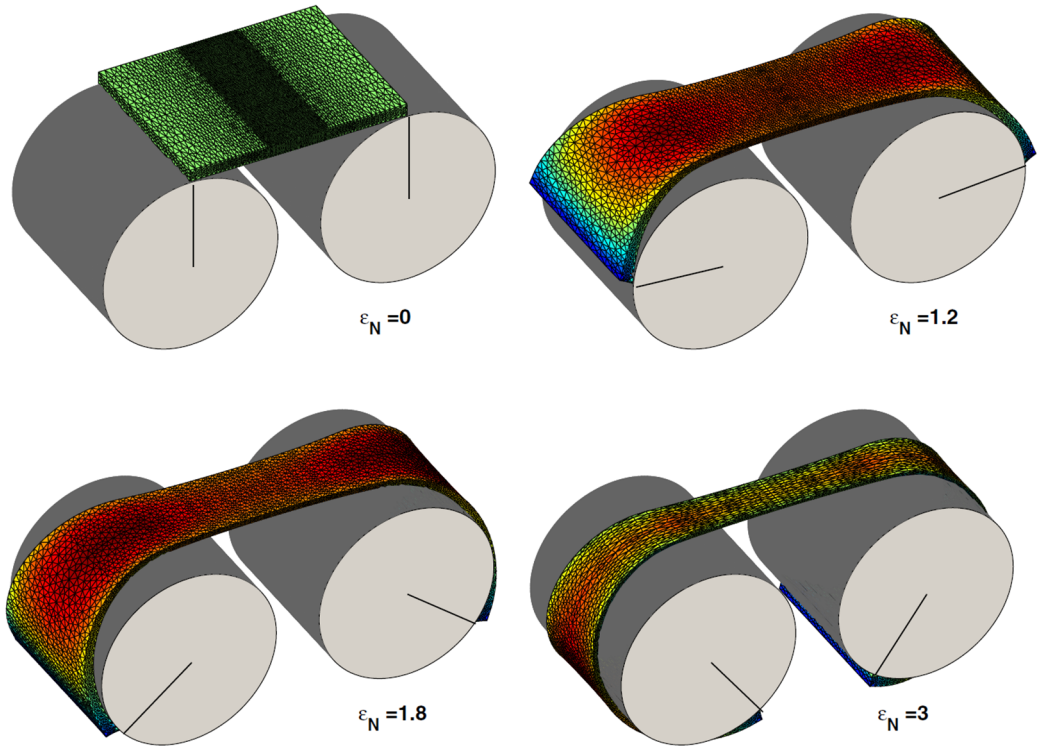


Figure 1.9: Elongation of a viscoelastic material in a Sentmanat extensional rheometer (Hassager et al. 2010).

When deformations are too large, the Lagrangian methods require a remeshing of the deformed geometry. Automatic *adaptive remeshing techniques* further extend the possibilities of the Lagrangian methods. As an example, a three-dimensional Lagrangian solver with adaptive remeshing has been developed by Rasmussen (1999) to compute the deformation of viscoelastic materials which are described with integral constitutive models. This algorithm has been used to simulate the extensional behavior of polymeric materials in a filament stretching rheometer (Rasmussen & Hassager 1999), and a Sentmanat extensional rheometer (Hassager et al. 2010), see Figure 1.9. However, the remeshing of the geometry increases in this case the computational cost as compared to the ALE method. In addition, the interpolation of the solution from the old mesh to the new mesh introduces additional numerical approximations.

Finally, a common drawback of all the Lagrangian methods is their difficulties to handle the change of topology in the geometry, produced for instance by interface splitting or interface merging. Thus, in spite of their flexibility, the Lagrangian methods may lack adaptability to simulate complex free-surface/interface dynamics problems.

1.2.2. Eulerian methods

The Eulerian methods are based on the opposite approach, where the material flows through a stationary computational mesh. Thus, the Eulerian methods require the evaluation of the mass and momentum fluxes between the elements (or control volumes) of the mesh. For this reason, Eulerian methods are typically coupled with a finite–volume discretization, which is directly based on the flux balances, but a finite–element method based on the variational formulation (or weak form) of the conservation laws can also be used. The Eulerian methods can be used with both structured meshes and unstructured meshes, which give more flexibility for the representation of complex geometries. In the context of free-surface flows, or multiphase flows of immiscible

materials, sharp interfaces are either represented explicitly with additional geometrical objects (i.e. markers or polygons), or implicitly through an additional field (i.e. the color function or the level-set function). A common advantage of the Eulerian methods with implicit interface representations is their ability to treat interface splitting and interface merging naturally, without the need of complex geometrical operations.

Historically, the first simulation of free surface flows used the *Marker-And-Cell* (MAC) method, where the position of the interface is represented by markers that are advected with the flow, similarly as Lagrangian particles. A complete review of the MAC method, with an emphasis on free-surface flows in computational rheology, has been presented by McKee et al. (2008). In particular, an improved MAC method, named GENSMAC, has been developed by Tomé et al. (Tome & McKee 1994; Tomé et al. 2001) to simulate three-dimensional flows of generalized Newtonian fluids (Tomé, Duffy & McKee 1996; Tomé et al. 2004), and viscoelastic materials on a Cartesian grid, see Figure 1.10. The GENSMAC method has been very efficient to simulate extrudate swelling (Mompean et al. 2011), jet buckling and container filling (Figueiredo et al. 2013) of polymeric materials, among others.

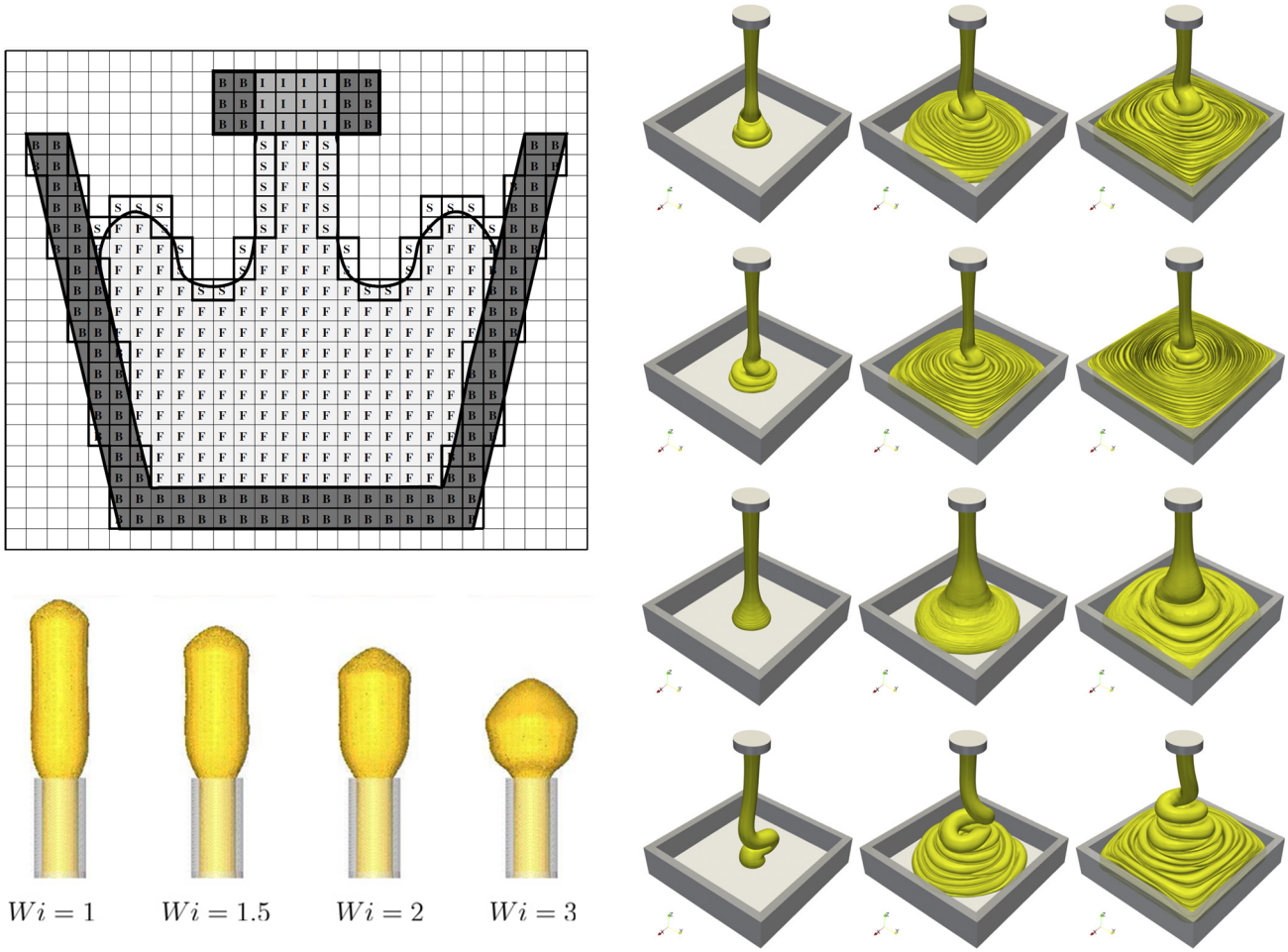


Figure 1.10: (Upper left) Schematics of the GENSMAC approach of Tomé (McKee et al. 2004); (lower left) simulation of extrudate swelling for a viscoelastic fluid (Mompean et al. 2011); (right) simulation of jet buckling and container filling for a branched polymer (Figueiredo et al. 2013).

Alternative Eulerian methods have been developed to avoid the explicit advection of virtual particles; the most popular being the *Volume-Of-Fluid* (VOF) method, and the *Level-Set* (LS) method. In both methods, the multiphase flow is tracked by an additional scalar variable, the color function. In the VOF method, the discrete values (the averages over the control volume) of the color function correspond to the volume fraction of the material inside the control volumes. Sharp interface are reconstructed from the color function, using interface reconstruction algorithms. Finally the volume fractions are advected with a geometrical advection scheme. The advantage of the VOF method is that it is directly based on the conservation of the volume (similar to the conservation of the mass in the case of incompressible fluids).

The color function used in the LS method corresponds to the distance to the interface. Thus, the level-set function has smooth variations, which make it possible to solve its advection with classical algebraic schemes (i.e. finite–differences, finite–volumes, finite–elements, etc.). The LS method avoids the calculation of the interface reconstruction, and directly provides the interface curvature (useful to compute surface tension). However, the main disadvantages of the LS method are that: the volume conservation is not enforced, and sharp interfaces tend to be smeared, resulting in a loss of accuracy. Hybrid particle–LS methods (Enright et al. 2002) and the coupled LS–VOF methods (Ménard, Tanguy & Berlemont 2007) have been developed to improve the conservation of the LS methods.

Examples of Eulerian simulations taken from the literature are shown in Figures 1.11–1.13. A common drawback for the Eulerian method is their inability to resolve sub-grid fluid structure, such as thin filaments, fluid droplets or bubbles. However, this limit can be coped with by the use of *dynamical mesh refinement* with hierarchical grids. Moreover, Cartesian grid can represent complex geometries with the *immersed boundary method*, see Figure 1.14. Finally, the combination of both dynamical mesh refinement with hierarchical Cartesian grids and the immersed boundary method could give a powerful method with enough adaptability to simulate complex free-surface/interface problems, while keeping relative simplicity; see for instance the simulation of jet atomization of Fuster et al. (2009) in Figure 1.15.

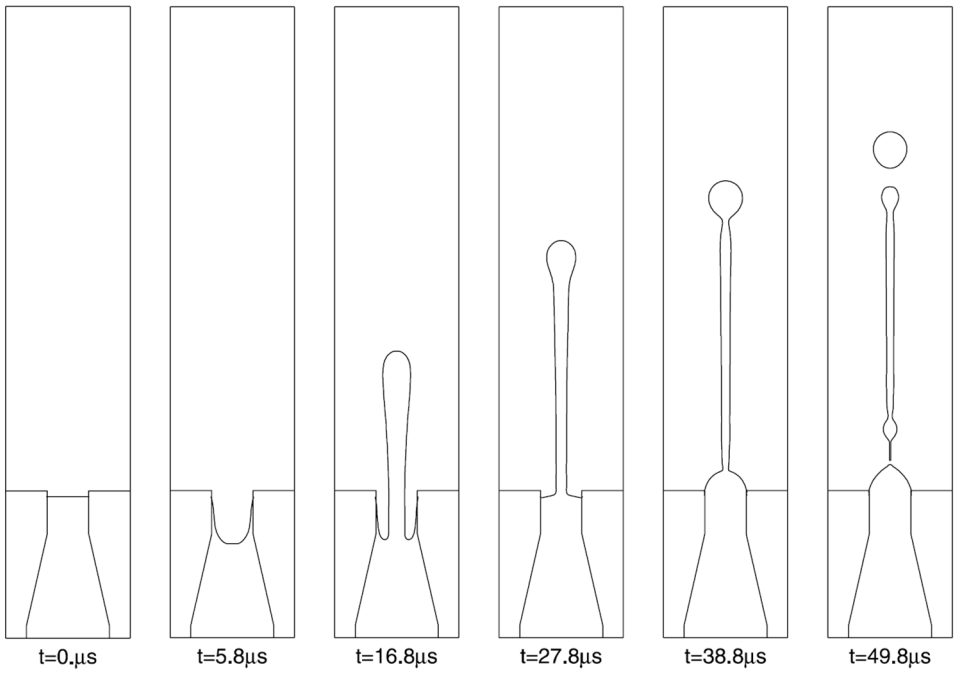


Figure 1.11: Simulation with the LS method of viscoelastic jet, application to ink jet printer (Yu, Sakai & Sethian 2007).

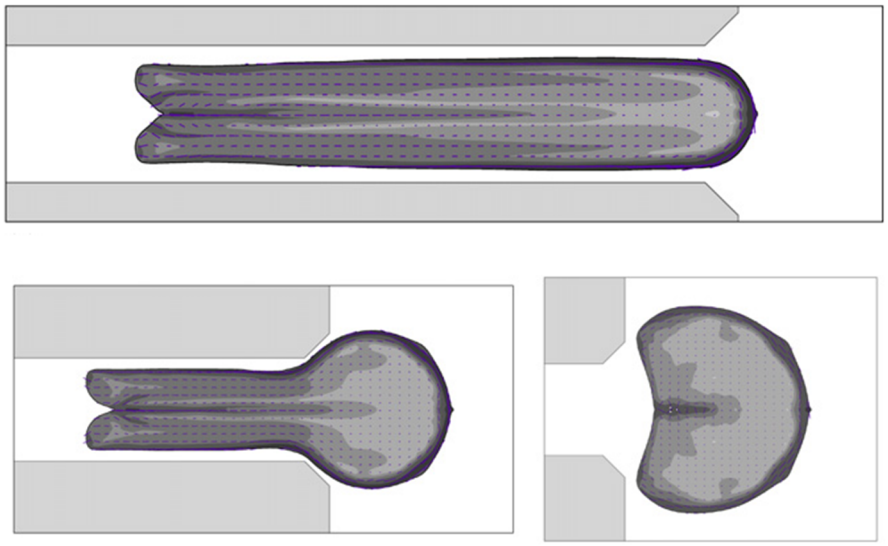


Figure 1.12: Simulation with the VOF method of a viscoelastic droplet passing through a microfluidic channel, application to the design of drug delivery system (Harvie, Cooper-White & Davidson 2008).

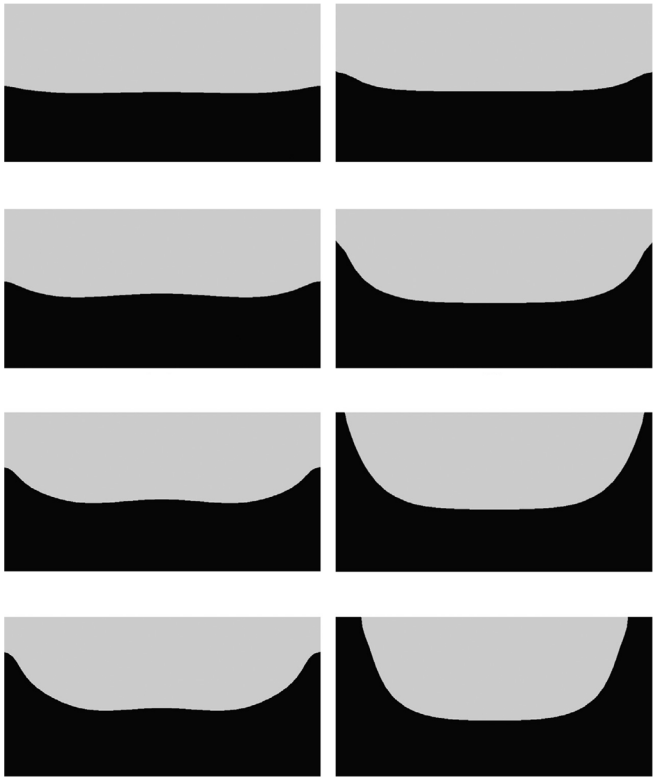


Figure 1.13: Simulation with an Eulerian method of encapsulation during the co-extrusion of viscoelastic fluids (Borzacchiello et al. 2013).

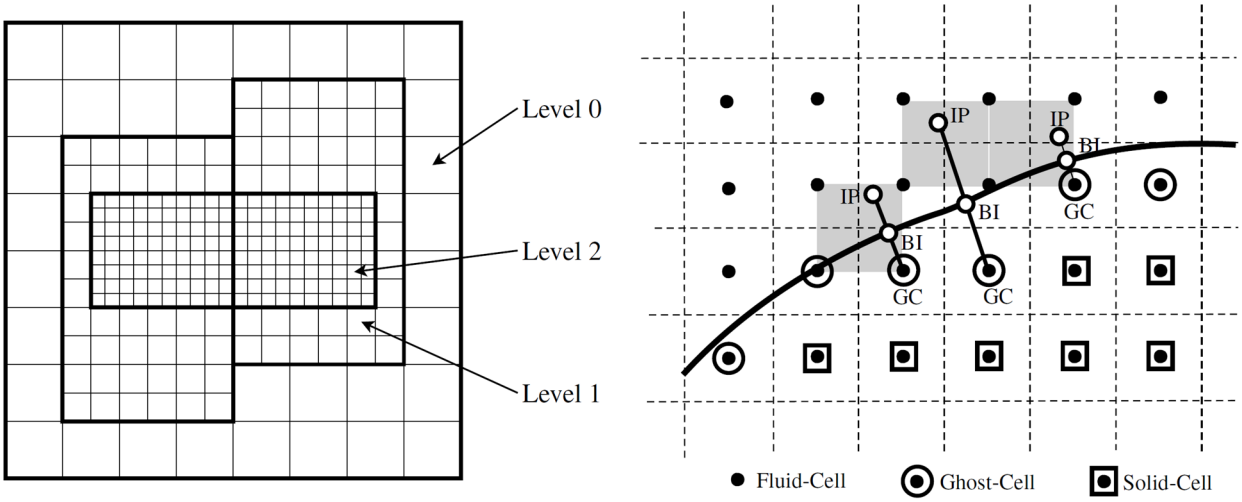


Figure 1.14: Enrichment of the Eulerian methods using Cartesian grids with (left) dynamical mesh refinement using hierarchical Cartesian grids (Griffith et al. 2007), and (right) the immersed boundary method (Mittal et al. 2008).

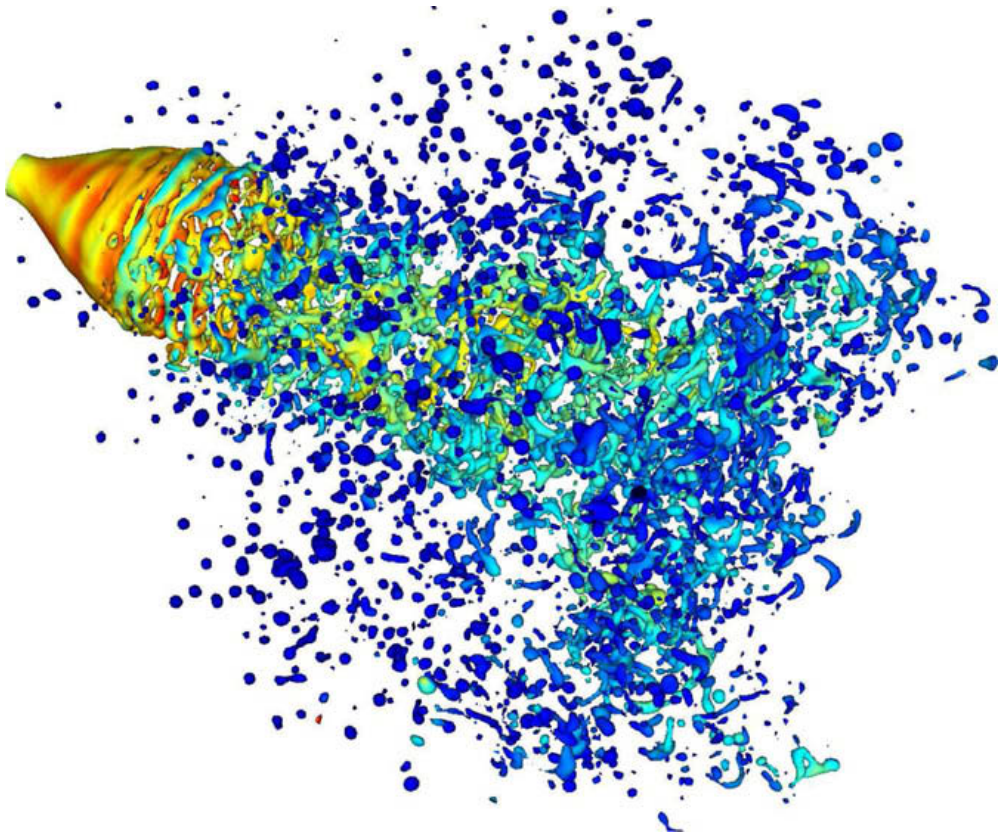


Figure 1.15: Simulation of the jet atomization of a Newtonian fluid, using the VOF method and a dynamical mesh refinement using hierarchical Cartesian grids (Fuster et al. 2009).

1.2.3. Discrete particle methods

The *discrete particle method* is a family of meshless numerical techniques, where the flow is represented through the movement of discrete particles rather than with a mesh. In contrast to Lagrangian and Eulerian methods, the discrete particle methods do not solve directly the conservation of physical quantities (i.e. mass, momentum and energy) in a continuum media, but rather solve the kinetics (i.e. the collisions) of discrete particles.

The *Smoothed-Particle-Hydrodynamics* (SPH) method is a discrete particle method, which was initially developed to the modeling of hydrodynamics flows. Moreover, it has also been used to simulate the deformation of solid object (Bonet 2004). Variants of the SPH method have been adapted to simulate free-surface flows of viscoelastic fluids (Fang et al. 2006; Rafiee, Manzari & Hosseini 2007; Xu & Ouyang 2013). However, stable SPH calculations of viscoelastic flows require artificial stress terms in order to avoid numerical instabilities (Fang et al. 2006). An advantage of the SPH method is its ability to represent naturally free surface flows, without the additional calculations due to the remeshing (in Lagrangian methods) or front tracking (in Eulerian methods). The SPH method is also adapted to simulate the mixing of miscible fluid, even though there is a limitation on the density ratio of the two phases. Nevertheless, accurate calculations require a large amount of discrete particles.

1.3. Current and future challenges

In spite of the variety of numerical solvers that have been developed, the simulation of viscoelastic flows is still a difficult task. Some flow problems can be addressed with several methods, see for instance the comparison work of Sujatha et al. (2006); however, Computational Rheology still lacks a unified approach that could deal with the entire range of problems, from pure solid mechanics to pure Newtonian fluid dynamics. The ideal method should have high adaptability, while keeping simplicity. However, these two qualities are usually in opposition. Consequently, it is often more efficient to use (or to develop) dedicated codes dealing only with a specific class of problems.

Many solvers suffer numerical stability issues, especially when the elastic component of the viscoelastic constitutive model has a dominant effect. A selected citation of Brown et al. (1993) from a paper published in *Theoretical and Computational Fluid Dynamics*, about the computation of viscoelastic flows with mixed finite-element methods, give an example of the encountered problems:

> Algorithms like the EEME/SUPG and EVSS/SUPG [...] are limited by numerical instabilities for time-dependent calculations. The instabilities are linked to the very difficult problem associated with accurate calculation of the eigenvalue spectrum for simple shear flow. The most dangerous eigenvalue has a real part that approaches $-1/2De$ and an imaginary part that scales as k, where k is the streamwise wave number, for $De \gg 1$; moreover, the eigenfunction develops boundary layer structure in the cross-stream direction. It is hard to imagine a more difficult problem to resolve. (Brown et al. 1993, p. 103)

A more recent citation of Morozov and van Saarloos (2007), from a paper published in *Physics Reports*, also summarizes the situation:

> If Newtonian fluid dynamics remains a challenge even today, then visco-elastic flows seem to pose unsurmountable problems. (Morozov & van Saarloos 2007, p. 115)

Indeed, one of the longstanding problems of Computational Rheology is the so-called *high Weissenberg number problem* (HWNP), which has been described in several review articles (Keunings 2000; Walters & Webster 2003). The HWNP manifests itself by a loss of reliability of the simulations, leading to a blowup of the numerical values. Hulsen, Fattal and Kupferman (2005) have shown evidences that the HWNP have a numerical origin.

Generally, numerical stability does not necessarily always go hand in hand with numerical accuracy. Very promising remedies of the HWNP, relying on changes of variables in the constitutive equations, have recently been proposed (Lozinski & Owens 2003; Fattal & Kupferman 2004; Fattal & Kupferman 2005; Balci et al. 2011). Moreover, the improvement of the current algorithms towards more robust, reliable and efficient numerical methods is still an ongoing research work.

In additional, to numerical instabilities, many of the popular constitutive models exhibit unphysical features (Rallison & Hinch 1988), referred to as *constitutive instabilities*. Kwon and Leonov (1995) classified the constitutive instabilities into two types: the *Hadamard instabilities* and the *dissipative instabilities*. The first is related to elastic material responses under sudden or high frequency wave disturbances. The latter is related to unbounded free energy (or rate of energy dissipation) with increasing deformations. Moreover, the distinction between purely numerical instabilities and purely constitutive instabilities is not always clear (Lee et al. 2004).

The development of realistic constitutive models is also a very active field of research. Knowledge about constitutive modelling can be gained from two complementary approaches: experimental rheology and theoretical physics. On the one hand, experimental rheology consists of testing the responses of complex fluids in simple rheometric flows, i.e. shear flows, extensional flows, etc. The extension of constitutive behaviors obtained by curve fitting, in order to describe more complex flows, should however be approached cautiously, because of the complex nature of the fluid. On the other hand, theoretical physics has proposed new constructive equations, based on idealizations of the molecular structure of the fluids (topology of the polymer chains, kinetics relations, etc.). In addition to idealizations, the derivation of constitutive equations often requires approximations (i.e. statistical averages, closure approximations, algebraic relations, etc.). Therefore, the constitutive models derived from molecular theories may not always give predictions in accordance with reality. Moreover, to date there are no theoretical models for complex fluids such as poly-dispersed fiber suspensions, colloids, emulsion, etc. Thus, the two approaches are complementary, and one must be aware of their limitations when choosing an appropriate constitutive model.

Finally, the applications dealing with complex fluids generally involve other physical phenomena, for instance heat transfers (e.g. in plastic processing), chemical reactions (e.g. curing of thermoplastic), electromagnetism (electro-active polymers), electro-osmotic reaction, etc. Therefore, Computational Rheology should also address the issues of *multiphysics* numerical simulations.

1.4. Aim and structure of the thesis

Among the different challenges listed in the previous section, this thesis aims at developing an efficient and reliable algorithm for two-dimensional simulations of multi-phase non-Newtonian fluids, in general, and bi-phasic/free-surface viscoelastic flows, in particular. The emphasis was put on potential applications in polymer processing. In particular, the extrusion and co-extrusion processes are very interesting applications, since they can exhibit viscoelastic dynamical instabilities, in spite of simple geometries.

Based on a critical review of the existing numerical methods, this thesis proposes a new framework within the family of the Eulerian methods. The developed methodology is based on the combination of three features: (i) a pressureless flow solver using a pure-streamfunction formulation, (ii) a viscoelastic stress solver for differential models reformulated with the log–conformation representation, and (iii) an interface tracking procedure using the volume-of-fluid method with a new cellwise conservative unsplit advection scheme.

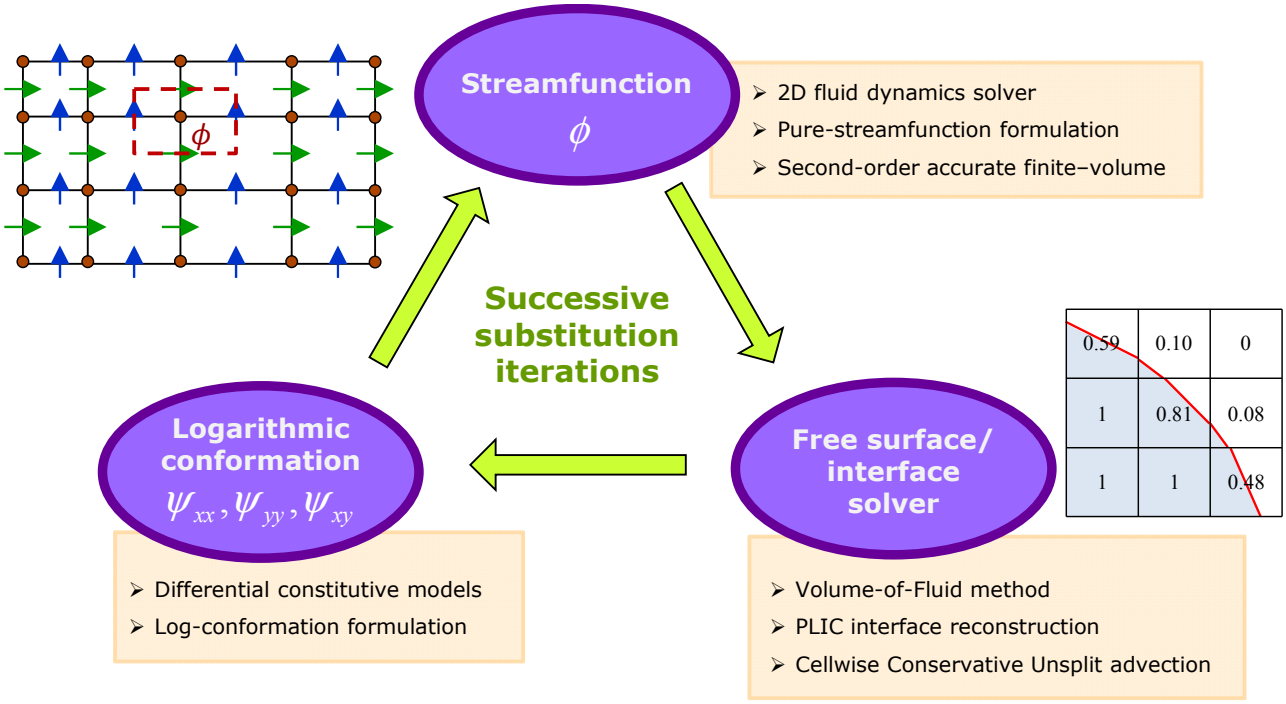


Figure 1.16: Schematic summary of the developed in-house Matlab code.

This numerical approach has been implemented in Matlab. A schematic summary of the developed in-house code is represented in Figure 1.16. The current algorithm is limited to two-dimensional geometries, but potential extensions to three-dimensional problems are discussed throughout the thesis.

Chapter 2 presents the non-Newtonian constitutive models of the polymeric materials. It includes empirical constitutive equations for inelastic (purely viscous) fluids and differential viscoelastic models derived from molecular theories of polymers. The numerical difficulties related to the HWNP together with its remedy, the log–conformation representation, are also summarized. Chapter 3 reviews the various formulations of the equations of motion, and in particular the pure streamfunction formulation. Chapter 4 describes the two-dimensional discretization of the streamfunction/log–conformation governing equations with the finite–volume method, and the solution algorithm of the viscoelastic flow solver. Chapter 5 deals with the modeling of immiscible multiphase flows with the VOF method. Piecewise linear interface reconstruction algorithms are reviewed, and two geometrical advection schemes are presented: the classical directional-split advection scheme and a new *Cellwise Conservative Unsplit* (CCU) scheme, which has been developed in the scope of this thesis. An alternative rule of mixtures, using the quadratic mean of the arithmetic and harmonic averages of the bulk material properties is also derived. Some examples of numerical simulations, in particular the lid-driven cavity, the planar 4:1 contraction, and the planar die exit geometries, are shown in chapter 6. Finally, the conclusions of the thesis and the perspectives for future works are summarized in chapter 7.

Four scientific papers written in the scope of this thesis are included in Appendices. Two of the appended papers have been published in the *Journal of Computational Physics* and the *Journal of Non-Newtonian Fluid Mechanics*, while the two others papers have been presented at international conferences.

1.5. References

Afonso, AM, Oliveira, PJ, Pinho, FT & Alves, MA 2011, 'Dynamics of high-Deborah-number entry flows: a numerical study', *Journal of Fluid Mechanics*, vol. 677, pp. 272–304.

Anderson, PD, Dooley, J & Meijer, HE 2006, 'Viscoelastic effects in multilayer polymer extrusion', *Applied Rheology*, vol. 16, 198–205.

Balci, N, Thomases, B, Renardy, M & Doering, CR 2011, 'Symmetric factorization of the conformation tensor in viscoelastic fluid models', *Journal of Non-Newtonian Fluid Mechanics*, vol. 166, pp. 546–553.

Bertola, V, Meulenbroek, B, Wagner, C, Storm, C, Morozov, A, van Saarloos, W & Bonn, D 2003, 'Experimental evidence for an intrinsic route to polymer melt fracture phenomena: A nonlinear instability of viscoelastic Poiseuille flow', *Physical Review Letters*, vol. 90, pp. 114502.

Bird, RB, Armstrong, RC & Hassager, O 1987, *Dynamics of polymeric liquids, Volume 1: fluid mechanics*, 2nd edn, John Wiley & Sons, New York.

Bonet, J, Kulasegaram, S, Rodriguez-Paz, MX & Profit, M 2004, 'Variational formulation for the smooth particle hydrodynamics (SPH) simulation of fluid and solid problems', *Computer Methods in Applied Mechanics and Engineering*, vol. 193, pp. 1245–1256.

Borzacchiello, D, Leriche, E, Blottière, B & Guillet, J 2013, 'Three-dimensional finite volume computation of viscoelastic fluid encapsulation by phase-field modeling', *Journal of Non-Newtonian Fluid Mechanics*, vol. 200, pp. 52–64.

Brasseur, E, Fyrillas, MM, Georgiou, GC & Crochet, MJ 1998, 'The time-dependent extrudate-swell problem of an Oldroyd-B fluid with slip along the wall', *Journal of Rheology*, vol. 42, pp. 549–566.

Brown, RA, Szady, MJ, Northey, PJ & Armstrong, RC 1993, 'On the numerical stability of mixed finite-element methods for viscoelastic flows governed by differential constitutive equations', *Theoretical and Computational Fluid Dynamics*, vol. 5, pp. 77–106.

Chen, K 1991, 'Interfacial instability due to elastic stratification in concentric coextrusion of two viscoelastic fluids', *Journal of Non-Newtonian Fluid Mechanics*, vol. 40, pp. 155–175.

Chen, KP & Zhang, Y 1993, 'Stability of the interface in co-extrusion flow of two viscoelastic fluids through a pipe', *Journal of Fluid Mechanics*, vol. 247, pp. 489–502.

Crochet, MJ & Keunings, R 1980, 'Die swell of a Maxwell fluid: numerical prediction', *Journal of Non-Newtonian Fluid Mechanics*, vol. 7, pp. 199–212.

Crochet, MJ & Keunings, R 1982, 'Finite element analysis of die swell of a highly elastic fluid', *Journal of Non-Newtonian Fluid Mechanics*, vol. 10, pp. 339–356.

Crochet, MJ & Walters, K 1993, 'Computational rheology: a new science', *Endeavour*, vol. 17, pp. 64–77.

Demay, Y & Agassant, JF 2014, 'An overview of molten polymer drawing instabilities', *International Polymer Processing*, vol. 29, pp. 128–139.

Denn, MM 2001, 'Extrusion instabilities and wall slip', *Annual Review of Fluid Mechanics*, vol. 33, 265–287.

Denn, MM 2004, 'Fifty years of non-Newtonian fluid dynamics', *AIChE Journal*, vol. 50, pp. 2335–2345.

Dooley, J & Rudolph, L 2003, 'Viscous and elastic effects in polymer coextrusion', *Journal of Plastic Film and Sheeting*, vol. 19, 111–122.

Enright, D, Fedkiw, R, Ferziger, J & Mitchell, I 2002, 'A hybrid particle level set method for improved interface capturing', *Journal of Computational Physics*, vol. 183, pp. 83–116.

Étienne, J, Hinch, EJ & Li, J 2006, 'A Lagrangian–Eulerian approach for the numerical simulation of free-surface flow of a viscoelastic material', *Journal of Non-Newtonian Fluid Mechanics*, vol. 136, pp. 157–166.

Fang, J, Owens, RG, Tacher, L & Parriaux, A 2006, 'A numerical study of the SPH method for simulating transient viscoelastic free surface flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 139, pp. 68–84.

Fattal, R & Kupferman, R 2004, 'Constitutive laws for the matrix-logarithm of the conformation tensor', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 281–285.

Fattal, R & Kupferman, R 2005, 'Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 126, pp. 23–37.

Figueiredo, RA, Oishi, CM, Cuminato, JA & Alves, MA 2013, 'Three-dimensional transient complex free surface flows: Numerical simulation of XPP fluid', *Journal of Non-Newtonian Fluid Mechanics*, vol. 195, pp. 88–98.

Fuster, D, Bagué, A, Boeck, T, Le Moyne, L, Leboissetier, A, Popinet, S, Ray, P, Scardovelli R & Zaleski, S 2009, 'Simulation of primary atomization with an octree adaptive mesh refinement and VOF method', *International Journal of Multiphase Flow*, vol. 35, pp. 550–565.

Georgiou, GC & Crochet, MJ 1994, 'Time-dependent compressible extrudate-swell problem with slip at the wall', *Journal of Rheology*, vol. 38, pp. 1745–1755.

Georgiou, GC 1996, 'On the stability of the shear flow of a viscoelastic fluid with slip along the fixed wall', *Rheologica Acta*, vol. 35, pp. 39–47.

Georgiou, GC 2003, 'The time-dependent, compressible Poiseuille and extrudate-swell flows of a Carreau fluid with slip at the wall', *Journal of Non-Newtonian Fluid Mechanics*, vol. 109, pp. 93–114.

Graham, MD 2004, 'Drag reduction in turbulent flow of polymer solutions', *Rheology Reviews*, vol. 2, pp. 143–170.

Griffith, BE, Hornung, RD, McQueen, DM & Peskin, CS 2007, 'An adaptive, formally second order accurate version of the immersed boundary method', *Journal of Computational Physics*, vol. 223, pp. 10–49.

Groisman, A & Steinberg, V 2000, 'Elastic turbulence in a polymer solution flow', *Nature*, vol. 405, pp. 53–55.

Han, CD 1973, 'A study of bicomponent coextrusion of molten polymers', *Journal of Applied Polymer Science*, vol. 17, pp. 1289–1303.

Harvie, DJE, Cooper-White, JJ & Davidson, MR 2008, 'Deformation of a viscoelastic droplet passing through a microfluidic contraction', *Journal of Non-Newtonian Fluid Mechanics*, vol. 155, pp. 67–79.

Hassager, O, Marin, JMR, Yu, K & Rasmussen, HK 2010, 'Polymeric liquids in extension: fluid mechanics or rheometry?', *Rheologica Acta*, pp. 49, vol. 543–554.

Hatzikiriakos, SG & Dealy, JM 1992, 'Role of slip and fracture in the oscillating flow of HDPE in a capillary', *Journal of Rheology*, vol. 36, pp. 845–884.

Hatzikiriakos, SG 2012, 'Wall slip of molten polymers', *Progress in Polymer Science*, vol. 37, pp. 624–643.

Hinch, EJ, Harris, OJ & Rallison, JM 1992, 'The instability mechanism for two elastic liquids being co-extruded', *Journal of Non-Newtonian Fluid Mechanics*, vol. 43, pp. 311–324.

Hoy, CV, Barda, A, Griffith, M & Halloran, JW 1998, 'Microfabrication of Ceramics by Co-extrusion', *Journal of the American Ceramic Society*, vol. 81, pp. 152–158.

Hulsen, MA, Fattal, R & Kupferman, R 2005, 'Flow of viscoelastic fluids past a cylinder at high Weissenberg number: stabilized simulations using matrix logarithms', *Journal of Non-Newtonian Fluid Mechanics*, vol. 127, pp. 27–39.

Ismael, MR, Clemens, F, Bohac, WM, Graule, T & Hoffmann, MJ 2009, 'Effects of rheology on the interface of $Pb(Zr,Ti)O_3$ monofilament composites obtained by co-extrusion', *Journal of the European Ceramic Society*, vol. 29, pp. 3015–3021.

Joseph, DD, Renardy, M & Renardy, Y 1984, 'Instability of the flow of two immiscible liquids with different viscosities in a pipe', *Journal of Fluid Mechanics*, vol. 141, pp. 309–317.

Jung, HW & Hyun, JC 2006, 'Instabilities in extensional deformation polymer processing', *Rheology Reviews*, vol. 4, pp. 131–164.

Keunings, R 2000, 'A survey of computational rheology', in: *Proceedings of the XIIIth International Congress on Rheology*, pp. 7–14, The British Society of Rheology, Cambridge.

Khomami, B & Ranjbaran, MM 1997, 'Experimental studies of interfacial instabilities in multilayer pressure-driven flow of polymeric melts', *Rheologica Acta*, vol. 36, pp. 345–366.

Khomami, B & Wilson, GM 1995, 'An experimental investigation of interfacial instability in superposed flow of viscoelastic fluids in a converging/diverging channel geometry', *Journal of Non-Newtonian Fluid Mechanics*, vol. 58, pp. 47–65.

Kwon, Y & Leonov, AI 1995, 'Stability constraints in the formulation of viscoelastic constitutive equations', *Journal of Non-Newtonian Fluid Mechanics*, vol. 58, pp. 25–46.

Larson, RG 1992, 'Instabilities in viscoelastic flows', *Rheologica Acta*, vol. 31, pp. 213-263.

Lee, J, Yoon, S, Kwon, Y & Kim, S 2004, 'Practical comparison of differential viscoelastic constitutive equations in finite element analysis of planar 4: 1 contraction flow', *Rheologica Acta*, vol. 44, pp. 188–197.

Li, X-B, Li, F-C, Cai, W-H, Zhang, H-N & Yang, J-C 2012, 'Very-low-Re chaotic motions of viscoelastic fluid and its unique applications in microfluidic devices: A review', *Experimental Thermal and Fluid Science*, vol. 39, pp. 1–16.

Lozinski, A & Owens, RG 2003, 'An energy estimate for the Oldroyd B model: theory and applications', *Journal of Non-Newtonian Fluid Mechanics*, vol. 112, pp. 161–176.

McKee, S, Tomé, MF, Cuminato, JA, Castelo, A & Ferreira, VG 2004, 'Recent advances in the marker and cell method', *Archives of Computational Methods in Engineering*, vol. 11, pp. 107–142.

Ménard, T, Tanguy, S & Berlemont, A 2007, 'Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet', *International Journal of Multiphase Flow*, vol. 33, pp. 510–524.

Meulenbroek, B, Storm, C, Bertola, V, Wagner, C, Bonn, D & van Saarloos, W 2003, 'Intrinsic route to melt fracture in polymer extrusion: a weakly nonlinear subcritical instability of viscoelastic Poiseuille flow', *Physical Review Letters*, vol. 90, pp. 024502.

Mittal, R, Dong, H, Bozkurttas, M, Najjar, FM, Vargas, A & von Loebbecke, A 2008, 'A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries', *Journal of Computational Physics*, vol. 227, pp. 4825–4852.

Mompean, G, Thais, L, Tomé, MF & Castelo, A 2011, 'Numerical prediction of three-dimensional time-dependent viscoelastic extrudate swell using differential and algebraic models', *Computers & Fluids*, vol. 44, pp. 68–78.

Morozov, AN & van Saarloos, W 2007, 'An introductory essay on subcritical instabilities and the transition to turbulence in visco-elastic parallel shear flows', *Physics Reports*, vol. 447, pp. 112–143.

Nigen, S, El Kissi, N, Piau, JM & Sadun, S 2003, 'Velocity field for polymer melts extrusion using particle image velocimetry: stable and unstable flow regimes'. *Journal of Non-Newtonian Fluid Mechanics*, vol. 112, pp. 177–202.

Normandin, M, Clermont, JR, Guillet, J & Raveyre, C 1999, 'Three-dimensional extrudate swell experimental and numerical study of a polyethylene melt obeying a memory-integral equation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 87, pp. 1–25.

Piau, JM, El Kissi, N & Tremblay, B 1990, 'Influence of upstream instabilities and wall slip on melt fracture and sharkskin phenomena during silicones extrusion through orifice dies', *Journal of Non-Newtonian Fluid Mechanics*, vol. 34, pp. 145–180.

Piau, JM, Nigen, S & El Kissi, N 2000, 'Effect of die entrance filtering on mitigation of upstream instability during extrusion of polymer melts', *Journal of Non-Newtonian Fluid Mechanics*, vol. 91, pp. 37–57.

Pinarbasi, A & Liakopoulos, A 1995, 'Stability of two-layer Poiseuille flow of Carreau-Yasuda and Bingham-like fluids', *Journal of Non-Newtonian Fluid Mechanics*, vol. 57, pp. 227–241.

Rafiee, A, Manzari, MT & Hosseini, M 2007, 'An incompressible SPH method for simulation of unsteady viscoelastic free-surface flows', *International Journal of Non-Linear Mechanics*, vol. 42, pp. 1210–1223.

Rallison, JM & Hinch, EJ 1988, 'Do we understand the physics in the constitutive equation?', *Journal of Non-Newtonian Fluid Mechanics*, vol. 29, pp. 37–55.

Rasmussen, HK & Hassager, O 1999, 'Three-dimensional simulations of viscoelastic instability in polymeric filaments', *Journal of Non-Newtonian Fluid Mechanics*, vol. 82, pp. 189–202.

Rasmussen, HK 1999, 'Time-dependent finite-element method for the simulation of three-dimensional viscoelastic flow with integral models', *Journal of Non-Newtonian Fluid Mechanics*, vol. 84, pp. 217–232.

Shaqfeh, ESG 1996, 'Purely elastic instabilities in viscometric flows', *Annual Review of Fluid Mechanics*, vol. 28, pp. 129–185.

Shore, JD, Ronis, D, Piché, L & Grant, M 1997, 'Theory of melt fracture instabilities in the capillary flow of polymer melts', Physical Review E, vol. 55, pp. 2976.

Southern, JH & Ballman, R 1975, 'Additional observations on stratified bicomponent flow of polymer melts in a tube', *Journal of Polymer Science: Polymer Physics Edition*, vol. 13, pp. 863–869.

Sujatha, KS, Matallah, H, Banaai, MJ & Webster, MF 2006, 'Computational predictions for viscoelastic filament stretching flows: ALE methods and free-surface techniques (CM and VOF)', *Journal of Non-Newtonian Fluid Mechanics*, vol. 137, pp. 81–102.

Taliadorou, E, Georgiou, GC & Alexandrou, AN 2007, 'A two-dimensional numerical study of the stick–slip extrusion instability', *Journal of Non-Newtonian Fluid Mechanics*, vol. 146, pp. 30–44.

Tome, MF & McKee, S 1994, 'GENSMAC: A computational marker and cell method for free surface flows in general domains', *Journal of Computational Physics*, vol. 110, pp. 171–186.

Tomé, MF, Duffy, B & McKee, S 1996, 'A numerical technique for solving unsteady non-Newtonian free surface flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 62, pp. 9–34.

Tomé, MF, Cuminato, JA, Mangiavacchi, N & McKee, S 2001, 'GENSMAC3D: a numerical method for solving unsteady three-dimensional free surface flows', *International Journal for Numerical Methods in Fluids*, vol. 37, pp. 747–796.

Tomé, MF, Grossi, L, Castelo, A, Cuminato, JA, Mangiavacchi, N, Ferreira, VG, de Sousa, FS & McKee, S 2004, 'A numerical method for solving three-dimensional generalized Newtonian free surface flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 85–103.

Walters, K & Webster, MF 2003, 'The distinctive CFD challenges of computational rheology', *International Journal for Numerical Methods in Fluids*, vol. 43, pp. 577–596.

Wang, SQ, Drda, PA & Inn, YW 1996, 'Exploring molecular origins of sharkskin, partial slip, and slope change in flow curves of linear low density polyethylene', *Journal of Rheology*, vol. 40, pp. 875–898.

Wilson, GM & Khomami, B 1992, 'An experimental investigation of interfacial instabilities in multilayer flow of viscoelastic fluids: Part I. Incompatible polymer systems', *Journal of Non-Newtonian Fluid Mechanics*, vol. 45, pp. 355–384.

Wilson, GM & Khomami, B 1993, 'An experimental investigation of interfacial instabilities in multilayer flow of viscoelastic fluids. Part II. Elastic and nonlinear effects in incompatible polymer systems', *Journal of Rheology*, vol. 37, pp. 315–339.

Xu, X & Ouyang, J 2013, 'A SPH-based particle method for simulating 3D transient free surface flows of branched polymer melts', *Journal of Non-Newtonian Fluid Mechanics*, vol. 202, pp. 54–71.

Yarin, AL & Graham, MD 1998, 'A model for slip at polymer/solid interfaces', *Journal of Rheology*, vol. 42, pp. 1491–1504.

Yiantsios, SG & Higgins, BG 1988, 'Linear stability of plane Poiseuille flow of two superposed fluids', *Physics of Fluids*, vol. 31, pp. 3225.

Yih, CS 1967, 'Instability due to viscosity stratification', *Journal of Fluid Mechanics*, vol. 27, pp. 337–352.

You, R, Borhan, A & Haj-Hariri, H 2008, 'A finite volume formulation for simulating drop motion in a viscoelastic two-phase system', *Journal of Non-Newtonian Fluid Mechanics*, vol. 153, pp. 109–129.

Yu, JD, Sakai, S & Sethian, JA 2007, 'Two-phase viscoelastic jetting', *Journal of Computational Physics*, vol. 220, pp. 568–585.

Yue, P, Zhou, C, Dooley, J & Feng, JJ 2008, 'Elastic encapsulation in bicomponent stratified flow of viscoelastic fluids', *Journal of Rheology*, vol. 52, pp. 1027–1042.

Chapter 2

Constitutive modeling

The constitutive modeling provides the relations between the deformations and the stress response of the material. As mentioned in the introduction, non-Newtonian constitutive models are necessary to capture the complexity of the actual physics that produces flow instabilities in many polymer processes. The non-Newtonian constitutive modeling is the specificity of *Computational Rheology* that makes it a singular subject in the field of *Computational Fluid Dynamics*.

Two classes of non-Newtonian constitutive models are presented hereinafter: the *generalized Newtonian fluid* models, and the *differential viscoelastic fluid* models. The generalized Newtonian fluid model provides a general framework to the description of inelastic fluids, where the stress response only depends on the instantaneous rate-of-deformations. These fluid models approximate the dynamics of polymeric liquids, without solving additional partial-differential or integral equations. In particular, they are able to represent power-law and/or viscoplastic fluids. The differential viscoelastic fluid models describe the material behaviors where the stress response does not only depend on the instantaneous deformation, but also on the history of the deformations. Thus, the viscoelastic fluid models are able to take into account transient effects, where the deformation of the fluid depends on the specific time-scale of the flow (or the time-scale of observation). In this chapter as well as in the numerical experiments presented later in this thesis, we consider the partial-differential viscoelastic models, where deformations are expressed in terms of a strain rate tensor (i.e. spatial velocity gradients). The integral models, where deformations are formulated in terms of a strain tensor, are omitted. The numerical difficulties associated with the solving of differential viscoelastic models (i.e. the HWNP), and their possible remedies with changes of variables, such as the log–conformation representation, are discussed at end of this chapter.

2.1. Generalized Newtonian fluid models

The generalized Newtonian fluid models describe the behavior of purely viscous (inelastic) fluids, where the stress response $\boldsymbol{\tau}$ only depends on the instantaneous rate of deformations. Generalized Newtonian fluids are governed by the generic constitutive model (Bird, Armstrong & Hassager 1987):

$$\boldsymbol{\tau} = 2\eta_{\text{eff}}\boldsymbol{D}, \tag{2.1}$$

where

$$\boldsymbol{D} = \frac{1}{2}\left(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^{\mathrm{T}}\right) \tag{2.2}$$

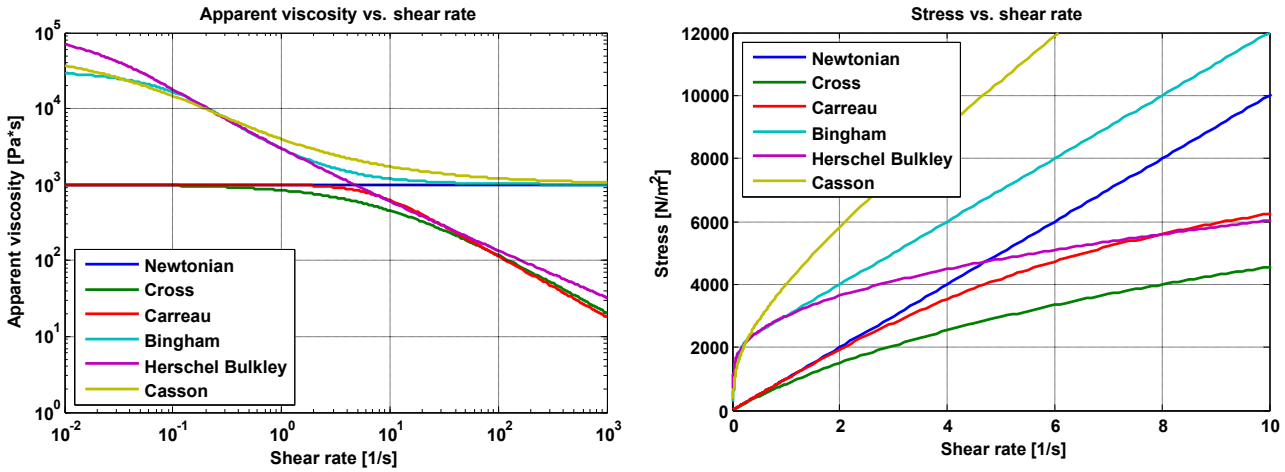


Figure 2.1: (left) Examples of viscosity functions for the generalized Newtonian fluid model, and (right) corresponding stress–strain-rate characteristic relations in simple shear flows. (The viscoplastic models are regularized.)

is the rate of deformation (or strain-rate) tensor, and η_{eff} is the effective (or apparent) viscosity, which can also depend on the magnitude of the rate of deformation:

$$\dot{\gamma} = \sqrt{2\{\boldsymbol{D} : \boldsymbol{D}\}} = \sqrt{2\sum_i \sum_j D_{ij}^{\,2}} \,. \tag{2.3}$$

Examples of effective viscosity functions $\eta_{\text{eff}}(\dot{\gamma})$ of generalized Newtonian fluid models, as well as the corresponding shear stress–shear rate characteristic relations in simple shear flows, are plotted in Figure 2.1.

2.1.1. Power-law fluids

The generalized Newtonian fluid model is able to describe power-law behavior of the fluid. In the power-law model, the shear stress τ_{xy} is given by

$$\tau_{xy} = K\dot{\gamma}^{n}, \tag{2.4}$$

where K is a consistency index, and n is the power-law index. The power-law model describes *shear-thinning* (pseudo-plastic) behaviors when $n < 1$, and *shear-thickening* (dilatant) behaviors when $n > 1$. The power-law behaviors come from microstructural changes of the material, due to the deformation. For instance, the shear-thinning behavior of polymeric solutions arises from the alignment of the polymer chains in the direction of the shear deformation at moderate to large shear rates, which result in less resistance to the flow than the initial situation with randomly orientated polymer chains. However, the power-law model leads to computational difficulties, at low shear-rates, since it produces very large apparent viscosities $\eta_{\text{eff}} \equiv \tau_{xy} / \dot{\gamma} = K\dot{\gamma}^{n-1}$ that eventually become indefinite when the material is at rest. Moreover, the material is not expected to experience microstructural changes when it is deformed at low shear-rates. For these two reasons, a plateau viscosity is desirable at low shear-rates. Therefore, in the general context of numerical simulations, where flows can be found within a wide range of deformation rates (including at a zero deformation rate), power-law fluids are rather governed by the Cross model or Carreau model, with the following apparent viscosity functions:

- Cross model:

$$\eta_{\text{eff}}(\dot{\gamma}) = \frac{\eta_0}{1+\left(\eta_0\dot{\gamma}/\tau^*\right)^{1-n}} \ . \tag{2.5}$$

- Carreau model:

$$\eta_{\text{eff}}(\dot{\gamma}) = \eta_\infty + (\eta_0 - \eta_\infty)\left(1+(\lambda\dot{\gamma})^2\right)^{\frac{n-1}{2}} \ . \tag{2.6}$$

These models circumvent the issues mentioned above and predict a plateau viscosity η_0 at low shear-rates. The transition between the plateau and the power-law regions is controlled by a characteristic shear-stress τ^* (in the Cross model) or a characteristic shear-rate $1/\lambda$ (in the Carreau model). In addition, the apparent viscosity of the Carreau model is limited by a lower bound η_∞ (which can be set to zero), for very large shear rates.

2.1.2. Viscoplastic fluids

Viscoplastic materials are able to flow only if they are submitted to stresses above the yield stress τ_{Y} that characterize them. Below the yield stress threshold, the viscoplastic materials behave like solids. The viscoplastic behavior is typically due to suspensions that creates weak solid-solid interactions. The popular models to describe the constitutive behavior of a viscoplastic fluid subject to shear deformations are (Mitsoulis 2007):

- Bingham model:

$$\begin{cases} \tau_{xy} = \tau_{\text{Y}} + \mu\dot{\gamma} & \text{for } \left|\tau_{xy}\right| > \tau_{\text{Y}}, \\ \dot{\gamma} = 0 & \text{for } \left|\tau_{xy}\right| \le \tau_{\text{Y}}. \end{cases} \tag{2.7}$$

- Herschel-Bulkley model:

$$\begin{cases} \tau_{xy} = \tau_{\text{Y}} + K\dot{\gamma}^n & \text{for } \left|\tau_{xy}\right| > \tau_{\text{Y}}, \\ \dot{\gamma} = 0 & \text{for } \left|\tau_{xy}\right| \le \tau_{\text{Y}}. \end{cases} \tag{2.8}$$

- Casson model:

$$\begin{cases} \sqrt{\tau_{xy}} = \sqrt{\tau_{\text{Y}}} + \sqrt{\mu\dot{\gamma}} & \text{for } \left|\tau_{xy}\right| > \tau_{\text{Y}}, \\ \dot{\gamma} = 0 & \text{for } \left|\tau_{xy}\right| \le \tau_{\text{Y}}. \end{cases} \tag{2.9}$$

In the Bingham model, the yielded viscoplastic fluid exhibits a constant plastic viscosity at large shear rates. In contrast, the Herschel-Bulkley model predicts a power-law behavior, which can either represent a shear-thinning or a shear-thickening behavior. The Casson model, which is popular in bio-mechanics, predicts a shear-thinning behavior that is different from the power-law behavior.

In the context of numerical simulation, we seek constitutive relations linking the stress tensor $\boldsymbol{\tau}$ to the rate of deformation tensor $\boldsymbol{D}$, without discontinuity in the apparent viscosity. The *Papanastasiou regularization* (Papanastasiou 1987) provides an efficient mean to approximate viscoplastic materials as generalized

Newtonian fluids. It consists in smoothing the transition from the unyielded (solid) region to the yielded (liquid) region using an inverse exponential function. The regularized Papanastasiou viscoplastic models have the following apparent viscosities functions:

- Regularized Papanastasiou Bingham model:

$$\eta_{\text{eff}}(\dot{\gamma}) = \frac{\tau_{\text{Y}}}{|\dot{\gamma}|}\left[1-\exp\left(-m|\dot{\gamma}|\right)\right] + \mu\,. \tag{2.10}$$

- Regularized Papanastasiou Herschel-Bulkley model:

$$\eta_{\text{eff}}(\dot{\gamma}) = \frac{\tau_{\text{Y}}}{|\dot{\gamma}|}\left[1-\exp\left(-m|\dot{\gamma}|\right)\right] + K\dot{\gamma}^{n-1}\,. \tag{2.11}$$

- Regularized Papanastasiou Casson model:

$$\eta_{\text{eff}}(\dot{\gamma}) = \left(\sqrt{\frac{\tau_{\text{Y}}}{|\dot{\gamma}|}}\left[1-\exp\left(-m|\dot{\gamma}|\right)\right] + \sqrt{\mu}\right)^2\,. \tag{2.12}$$

The limit between the unyielded and yielded region corresponds to the iso-surface where the magnitude of the stress tensor is equal to the yield stress:

$$|\boldsymbol{\tau}| = \tau_{\text{Y}}\,, \tag{2.13}$$

where the magnitude $|\boldsymbol{\tau}|$ of the stress tensor is given by:

$$|\boldsymbol{\tau}| = \sqrt{\frac{1}{2}\{\boldsymbol{\tau}:\boldsymbol{\tau}\}} = \sqrt{\frac{1}{2}\sum_i\sum_j \tau_{ij}^{\,2}}\,. \tag{2.14}$$

The regularization parameter *m* controls the smoothness of the transition from the unyielded to the yielded regions. The drawback of the regularized viscoplastic models is that they lose precision in the prediction of the yielded/unyielded limit. Lower values of the regularization parameter *m* increase the fidelity of the regularized model to the original viscoplastic models. However, numerical simulations can suffer convergence issues when the value of m is too small. Additional discussions concerning the accuracy of the regularized viscoplastic models, and usage recommendations are given by Frigaard and Nouar (2005).

2.2. Viscoelastic fluid models

The specificity of viscoelastic materials is that they have stress responses that do not only depend on their current deformations but also on their deformation history. Thus, time is a key parameter; and viscoelastic materials have different stress responses depending on the time-scale of the deformation. Viscoelastic materials typically exhibit stress relaxation, creep deformation and shape memory. The relative effect of viscoelasticity is usually quantified by the dimensionless *Weissenberg number Wi*, which is given by the product of a characteristic relaxation time λ of the material and a characteristic deformation rate $\dot{\varepsilon}$ of the flow:

$$Wi = \lambda\dot{\varepsilon}\,. \tag{2.15}$$

Another dimensionless quantity, the *Deborah number*, is sometimes used for flows with non-constant stretch history (i.e. deformation rate). It is defined as the ratio between the characteristic relaxation time and the time-scale of observation t_c:

$$De = \lambda/t_c\,. \tag{2.16}$$

In the next subsection, we specifically focus on the differential constitutive models derived from molecular models for polymers, where the viscoelasticity comes from the *kinetics* of the polymer chains. At the molecular scale, the relaxation of diluted short-chain polymers is captured by the Rouse model, while the kinetics of entangled long-chain and branched polymers are described in the tube theories, which include a reptation mechanism, see the reviews of Watanabe (1999) and McLeish (2002). However, the purpose of the constitutive models is the description of viscoelasticity at the scale of *continuum mechanics*.

2.2.1. Differential viscoelastic models

The derivation of continuum constitutive equations from the kinetics theories requires closure approximations. Nevertheless, in most numerical simulations of practical applications, the constitutive model and the material properties are rather determined from rheological measurements. It should be kept in mind that the extension of constitutive models estimated by curve-fitting of rheometric flows (e.g. in simple shear or uniaxial extensional), to the description of more complex flows, should be approached cautiously. Indeed, many of the popular constitutive models can predict unphysical features, referred to as *constitutive instabilities*, which can be classified into two types: the Hadamard instability and the dissipative instability (Kwon & Leonov 1995). The first is related to the elastic response under sudden or high frequency wave disturbances. The latter is related to unbounded free energy or rate of energy dissipation with increasing deformations. Thus, appropriate constitutive models should be chosen carefully.

A generic viscoelastic model, which encompasses most of the constitutive equations derived from molecular theories, is presented here. First, the stress response $\boldsymbol{\tau}$ to the deformations of the viscoelastic fluid is decomposed with the solvent–polymer stress splitting technique into a purely viscous contribution $\boldsymbol{\tau}_{\mathrm{S}}$, which corresponds to the instantaneous response of the solvent, and the polymer contribution $\boldsymbol{\tau}_{\mathrm{P}}$, accounting for the memory:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\mathrm{S}} + \boldsymbol{\tau}_{\mathrm{P}}, \tag{2.17}$$

with

$$\boldsymbol{\tau}_{\mathrm{S}} = 2\eta_{\mathrm{S}}\boldsymbol{D}, \tag{2.18}$$

$$\boldsymbol{\tau}_{\mathrm{P}} = G_0 f_{\mathrm{S}}(\boldsymbol{c}), \tag{2.19}$$

where $\eta_{\mathrm{S}} = \beta\eta_0$ is the solvent viscosity, $G_0 = (1-\beta)\eta_0/\lambda$ is the plateau elastic moduli, and $f_{\mathrm{S}}(\boldsymbol{c})$ is a strain function, which depends on the constitutive model. The steady-state viscosity η_0, the relaxation time λ, and the solvent viscosity ratio (or retardation ratio) $\beta \in [0,1]$ are the material parameters of linear viscoelasticity. In constitutive models derived from molecular theories, the strain function $f_{\mathrm{S}}(\boldsymbol{c})$ is expressed in terms of the conformation tensor $\boldsymbol{c}$, an internal tensorial variable which represents the macromolecular configuration of the polymer chains. The conformation tensor is defined as the statistical average of the dyadic product of the end-to-end connector vector $\vec{\boldsymbol{Q}}$ of the polymer chain building-up the elastic stress:

$$\boldsymbol{c} = \left\langle \vec{\boldsymbol{Q}}\vec{\boldsymbol{Q}}^{\mathrm{T}} \right\rangle \equiv \int \vec{\boldsymbol{Q}}\vec{\boldsymbol{Q}}^{\mathrm{T}} \psi\left(\vec{\boldsymbol{Q}}, t\right) d\vec{\boldsymbol{Q}}, \tag{2.20}$$

where $\psi\left(\vec{\boldsymbol{Q}}, t\right)$ is a probability distribution function following the Fokker–Planck equation. The end-to-end connector vector is defined in the molecular theory; it has different interpretations depending on the state of the polymer chain, i.e. a diluted polymer solution, a polymeric network, an entangled polymer melt, see Figure 2.2.

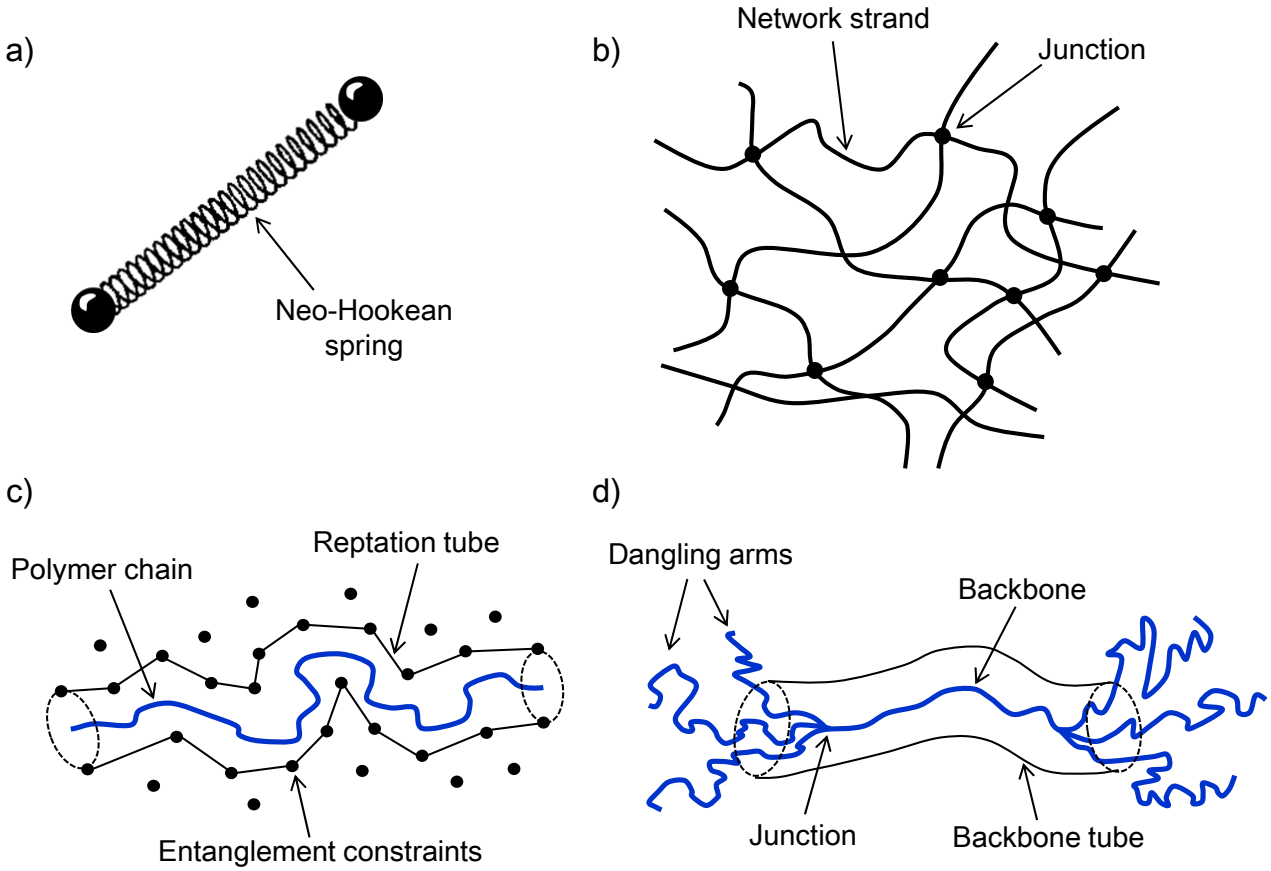


Figure 2.2: Molecular topology: (a) Dumbbell molecule in the FENE model, (b) Polymer network in the PTT model, (c) Entangled linear polymer in the Rolie–Poly model, (d) Pom–Pom molecule in the XPP model.

According to equation (2.20), the conformation tensor is by definition *symmetric positive definite*, independently of the kinetic theory. Moreover, the conformation tensor is scaled such that it is equal to the identity matrix $\boldsymbol{I}$ when the polymer chain is at the equilibrium.

Secondly, the basic mechanism of stress build-up and stress relaxation is governed by a differential equation in terms of the conformation tensor, with the generic form:

$$\overset{\triangledown}{\boldsymbol{c}} = -\frac{1}{\lambda} f_{\mathrm{R}}\left(\boldsymbol{c}\right), \tag{2.21}$$

where $\overset{\triangledown}{\boldsymbol{c}}$ is the upper-convected time derivative of the conformation tensor, λ is the relaxation time of the main relaxation mechanism (i.e. Brownian motion, reptation, retraction, etc.), and $f_{\mathrm{R}}\left(\boldsymbol{c}\right)$ is a relaxation function that depends on the constitutive model. The upper-convected time derivative is defined:

$$\overset{\triangledown}{\boldsymbol{c}} \equiv \frac{\partial \boldsymbol{c}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{c} - \left(\boldsymbol{c}\cdot\nabla\boldsymbol{u}^{\mathrm{T}} + \nabla\boldsymbol{u}\cdot\boldsymbol{c}\right), \tag{2.22}$$

where the contravariant terms (those which are inside the parenthesis) are necessary in order for the model to respect the *principle of material frame-invariance*, also referred to as material objectivity. In the constitutive models derived from molecular theories, the strain and relaxation functions, $f_{\mathrm{S}}\left(\boldsymbol{c}\right)$ and $f_{\mathrm{R}}\left(\boldsymbol{c}\right)$, are polynomials of the conformation tensor, whose coefficients can depend on the invariants of $\boldsymbol{c}$:

$$f(\boldsymbol{c}) = \sum_k g_k(I_c, II_c, III_c)\boldsymbol{c}^k, \tag{2.23}$$

where $I_c = \mathrm{tr}(\boldsymbol{c})$, $II_c = \left(\mathrm{tr}^2(\boldsymbol{c}) - \mathrm{tr}(\boldsymbol{c}^2)\right)/2$ and $III_c = \det(\boldsymbol{c})$ are the first, second and third invariants of $\boldsymbol{c}$, respectively. The strain function is generally a first-order polynomial, while the relaxation function is at most a second-order polynomial. Table 2.1 gives an overview of the strain and relaxation functions corresponding to the most popular constitutive models derived from molecular theories. These models are described below.

Moreover, the viscoelastic response of the material can be modelled with multiple modes:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_S + \sum_{i=1}^{m} G_i f_{S,i}(\boldsymbol{c}_i), \tag{2.24}$$

$$\overset{\nabla}{\boldsymbol{c}}_i = -\frac{1}{\lambda_i} f_{R,i}(\boldsymbol{c}_i). \tag{2.25}$$

where G_i, λ_i, $\boldsymbol{c}_i$, $f_{S,i}(\boldsymbol{c}_i)$, $f_{R,i}(\boldsymbol{c}_i)$ are relative to the i^{th} mode, and m is the total number of modes. Each mode describes the response of the material within a particular timescale, such that the entire range of the deformation rates is covered by the spectrum of the relation times λ_i. Multi-modal viscoelastic responses are due to the fact that real polymeric materials are generally composed of different polymers with various molecular weights. From a computational point of view, the calculation of the multi-modal viscoelastic response is parallelizable, since each viscoelastic mode is independent of the others.

Constitutive models	Strain function $f_S(\boldsymbol{c})$	Relaxation function $f_R(\boldsymbol{c})$
Oldroyd–B	$\boldsymbol{c} - \boldsymbol{I}$	$\boldsymbol{c} - \boldsymbol{I}$
FENE–P	$\dfrac{\boldsymbol{c}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2} - \boldsymbol{I}$	$\dfrac{\boldsymbol{c}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2} - \boldsymbol{I}$
FENE–CR	$\dfrac{\boldsymbol{c} - \boldsymbol{I}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2}$	$\dfrac{\boldsymbol{c} - \boldsymbol{I}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2}$
Giesekus	$\boldsymbol{c} - \boldsymbol{I}$	$\alpha\boldsymbol{c}^2 + (1-2\alpha)\boldsymbol{c} - (1-\alpha)\boldsymbol{I}$
Simplified PTT linear	$\boldsymbol{c} - \boldsymbol{I}$	$(1 + \varepsilon\,\mathrm{tr}(\boldsymbol{c} - \boldsymbol{I}))(\boldsymbol{c} - \boldsymbol{I})$
Simplified PTT exponential	$\boldsymbol{c} - \boldsymbol{I}$	$\exp(\varepsilon\,\mathrm{tr}(\boldsymbol{c} - \boldsymbol{I}))(\boldsymbol{c} - \boldsymbol{I})$
Rolie–Poly	$\boldsymbol{c} - \boldsymbol{I}$	$\left(1 + \varpi\left(1 - \sqrt{3/\mathrm{tr}\,\boldsymbol{c}}\right)\left(1 + \beta_{\mathrm{ccr}}(\mathrm{tr}\,\boldsymbol{c}/3)^{\delta}\right)\right)\boldsymbol{c} - \left(1 + \varpi\left(1 - \sqrt{3/\mathrm{tr}\,\boldsymbol{c}}\right)\beta_{\mathrm{ccr}}(\mathrm{tr}\,\boldsymbol{c}/3)^{\delta}\right)\boldsymbol{I}$
XPP	$\boldsymbol{c} - \boldsymbol{I}$	$\alpha\boldsymbol{c}^2 + \dfrac{1}{\Lambda^2}\left[1 - \alpha\left(1 + \dfrac{1}{3}\mathrm{tr}(\boldsymbol{c}^2)\right) + \varpi(\Lambda^2 - \Lambda)e^{2(\Lambda-1)/q}\right]\boldsymbol{c} - (1-\alpha)\boldsymbol{I}$ where $\Lambda = \sqrt{\mathrm{tr}(\boldsymbol{c})/3}$

Table 2.1: Summary of the strain and relaxation functions for popular constitutive models derived from molecular theories.

- Oldroyd–B model:

The Oldroyd–B model (or contravariant convected Jeffreys model), originally proposed by Oldroyd (1950), was the first material frame-invariant (respecting the principle of material objectivity) phenomenological model describing viscoelastic fluids. The Oldroyd–B model assumes linear viscoelasticity, with identical strain and relaxation functions:

$$f_S(\boldsymbol{c}) = f_R(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I} . \tag{2.26}$$

The Oldroyd–B model can be derived from the kinetic theory (Kramers 1946), assuming that polymer molecules behaving like a suspension of Hookean springs in a Newtonian solvent.

The Oldroyd–B model includes the basic feature of viscoelasticity: stress relation, creep deformations and a first normal stress difference. However, it only predicts a constant shear viscosity (like Boger fluids), and a zero second normal stress difference. In addition, the predictions in purely extensional flows are unrealistic, because of the absence of a mechanism that limits the extensibility of the linear springs. Under steady extensional deformations with a finite extensional rate $\dot{\varepsilon}$, the Oldroyd–B model gives rise to an infinite stress growth (and an infinite free energy), if the extensional rate is equal or larger than half of the reciprocal of the relaxation time: $\dot{\varepsilon} \geq \lambda^{-1}/2$. In spite of these drawbacks, the Oldroyd–B model remains a useful model to assess the reliability of numerical solvers, since it is the basis of most of the non-linear viscoelastic models further developed. In addition, the Oldroyd–B model generally gives rise to more numerical difficulties than the non-linear viscoelastic models, which have more realistic extensional behaviors. For these reasons, we have used the Oldroyd–B model in the numerical experiments presented later in this thesis.

- FENE model:

The family of the FENE (Finitely Extensible Nonlinear Elastic) models is adapted to describe the viscoelastic behavior of dilute polymeric solutions. The FENE models are based on the kinetic theory of polymer chains behaving like non-linear elastic springs. Warner (1972) calculated the constitutive response of a dilute monomer, approximated by the microstructural bead-string model with non-linear elastic connectors whose tension forces are given by:

$$F(Q) = \frac{H_0 Q}{1 - (Q/L)^2} , \tag{2.27}$$

where Q is the length of the string connector, H_0 is the spring constant, and L is the ultimate length of the connector. Bird, Dotson and Johnson (1980) derived the FENE–P constitutive model from the dumbbell FENE model, using the statistical average proposed by Peterlin (1966) as a closure approximation. The FENE–P model possesses a finite extensibility and predicts a shear-thinning behavior. The closure approximation was further modified by Chilcott and Rallison (1988) in order to eliminate the shear-rate dependency of the steady-state viscosity. Thus, the resulting constitutive model, referred to as the FENE–CR model, describes Boger fluids. The strain and relaxation functions are identical to each other, both in the FENE–P and FENE–CR. For the FENE–P model:

$$f_S(\boldsymbol{c}) = f_R(\boldsymbol{c}) = \frac{\boldsymbol{c}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2} - \boldsymbol{I} . \tag{2.28}$$

For the FENE–CR model:

$$f_S(\boldsymbol{c}) = f_R(\boldsymbol{c}) = \frac{\boldsymbol{c} - \boldsymbol{I}}{1 - \mathrm{tr}(\boldsymbol{c})/L^2} . \tag{2.29}$$

- Giesekus model:

Giesekus (1982) derived a constitutive model for concentrated polymer solutions with an inclusion of quadratic terms of the stress, which can be interpreted as the effect of anisotropic hydrodynamic drag induced by the polymer–polymer interactions (Bird & Wiest 1985). The corresponding strain and relaxation functions are:

$$\begin{cases} f_{\mathrm{S}}(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I}, \\ f_{\mathrm{R}}(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I} + \alpha(\boldsymbol{c} - \boldsymbol{I})^2. \end{cases} \tag{2.30}$$

The dimensionless parameter $0 \le \alpha \le 0.5$ is the mobility factor, which quantifies the effect of the anisotropic stretching.

- PTT model:

The PTT model has been derived by Phan-Thien and Tanner (1977; 1978) from the kinetic theory of an elastic network representing a polymeric melt. The model assumes that the rates of creation and destruction of junctions between the polymer strands depend on the average extension of the network. In the first version of the PTT model (Phan-Thien and Tanner, 1977), the relaxation function is assumed to have a linear dependence on the trace of the conformation tensor:

$$\begin{cases} f_{\mathrm{S}}(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I}, \\ f_{\mathrm{R}}(\boldsymbol{c}) = \big(1 + \varepsilon \operatorname{tr}(\boldsymbol{c} - \boldsymbol{I})\big)(\boldsymbol{c} - \boldsymbol{I}). \end{cases} \tag{2.31}$$

Later, Phan-Thien and Tanner (1978) derived a more accurate relaxation function:

$$\begin{cases} f_{\mathrm{S}}(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I}, \\ f_{\mathrm{R}}(\boldsymbol{c}) = \exp\big(\varepsilon \operatorname{tr}(\boldsymbol{c} - \boldsymbol{I})\big)(\boldsymbol{c} - \boldsymbol{I}). \end{cases} \tag{2.32}$$

The original PTT model includes the Gordon–Schowalter convected time derivative of the conformation tensor, which is a linear combination of the lower and upper convected time-derivative:

$$\overset{\circ}{\boldsymbol{c}} \equiv -\left(\frac{1+a}{2}\right)\left(\boldsymbol{c}\cdot\nabla\boldsymbol{u}^{\mathrm{T}} + \nabla\boldsymbol{u}\cdot\boldsymbol{c}\right) + \left(\frac{1-a}{2}\right)\left(\boldsymbol{c}\cdot\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^{\mathrm{T}}\cdot\boldsymbol{c}\right), \tag{2.33}$$

where a is a scalar parameter in the interval [–1; 1]. The upper and lower convected time derivative are recovered when $a = 1$ and $a = -1$, respectively. The Gordon–Schowalter derivative simulates non-affine displacements of the junctions with an effective slip, and $\xi = 1 - a$ was introduced as the slip parameter of the network. With two non-linear material parameters, ε and ξ, the PTT model had the ability to fit both the shear and extensional behavior of polymer melts. However, the PTT model is prone to the Hadamard instability when $\xi \neq 0$ (Kwon & Leonov 1995), which make it unsuitable for numerical simulations. The Hadamard instability is a mathematical artifact built-in the constitutive model, producing spurious oscillations when the model is subjected to sudden or high frequency wave disturbances, which should neither be confused with numerical instabilities (due to numerical approximations of the solver), nor the actual (physical) hydrodynamic instabilities. For this reason, we only present here the simplified PTT model with $\xi = 0$, where the Gordon–Schowalter convected time derivative corresponds to the upper convected time-derivative.

- Rolie–Poly model:

The Rolie–Poly model, standing for ROuse LInear Entangled POLYmers, (Likhtman & Graham 2003) is a differential constitutive equation derived for entangled linear (unbranched) polymers. The Rolie–Poly model is

based on the tube theory, and takes into account three phenomena: reptation, chain retraction, and convective constraint release. The lateral movement of the polymer chain is confined inside a tube, which represents the excluded volume of the neighboring entangled chains. The stochastic evolution of the tube orientations results in the following equation in term of the conformation tensor:

$$\overset{\nabla}{\boldsymbol{c}} = \underbrace{-\frac{1}{\lambda}(\boldsymbol{c}-\boldsymbol{I})}_{\text{Reptation}} \underbrace{-\frac{2}{\lambda_{\mathrm{R}}}\left(1-\sqrt{3/\operatorname{tr}\boldsymbol{c}}\right)\boldsymbol{c}}_{\text{Retraction}} \underbrace{-\frac{2\beta_{\mathrm{ccr}}\left(\operatorname{tr}\boldsymbol{c}/3\right)^{\delta}}{\lambda_{\mathrm{R}}}\left(1-\sqrt{3/\operatorname{tr}\boldsymbol{c}}\right)(\boldsymbol{c}-\boldsymbol{I})}_{\text{Convective constraint release}}, \tag{2.34}$$

where λ is the reptation time, λ_{R} is the retraction time, β_{ccr} is the convective constraint release parameter, and δ is the tube deformation parameter. Hence, the Rolie–Poly model is covered by the generic differential equation (2.21), where the strain and relaxation functions are:

$$\begin{cases} f_{\mathrm{S}}(\boldsymbol{c}) = \boldsymbol{c}-\boldsymbol{I}, \\ f_{\mathrm{R}}(\boldsymbol{c}) = \left(1+\varpi\left(1-\sqrt{3/\operatorname{tr}\boldsymbol{c}}\right)\left(1+\beta_{\mathrm{ccr}}\left(\operatorname{tr}\boldsymbol{c}/3\right)^{\delta}\right)\right)\boldsymbol{c} - \left(1+\varpi\left(1-\sqrt{3/\operatorname{tr}\boldsymbol{c}}\right)\beta_{\mathrm{ccr}}\left(\operatorname{tr}\boldsymbol{c}/3\right)^{\delta}\right)\boldsymbol{I}, \end{cases} \tag{2.35}$$

and $\varpi = 2\lambda/\lambda_{\mathrm{R}}$ is a material parameter characterizing the ratio of the reptation to the retraction time-scales.

- XPP model:

The XPP (eXtended Pom–Pom) model was introduced by Verbeeten, Peters and Baaijens (2001) as an improvement of the Pom–Pom model (McLeish & Larson 1998), for long-chain branched polymers. Both the Pom–Pom and the XPP models assume a molecular topology made of a backbone connecting two junction points of arm stars. The displacement of the backbone is constrained inside a tube, because of the excluded volume of the neighbouring entangled chains, while the dangling arms move freely outside the tube. The XPP model includes the following mechanisms: anisotropic relaxation of the backbone orientation (reptation), relaxation of the backbone stretch (retraction), and local branch-point withdrawal. The model considers that only the length of the polymer which is inside the tube participates to the stretching. Once the backbone length become shorter than the tube length, the arms segments are drawn into the tube, and also participate to the stretching, with a different relaxation time. The viscoelastic stress of the XPP model is given by:

$$\boldsymbol{\sigma}_e = G_0\left(3\Lambda^2\boldsymbol{S}-\boldsymbol{I}\right), \tag{2.36}$$

where G_0 is the linear elastic modulus, Λ is the backbone stretch (scalar value), and $\boldsymbol{S}$ is the orientation tensor of the backbone. The evolution of the backbone orientation is governed by the following equation:

$$\overset{\nabla}{\boldsymbol{S}} = -2\{\boldsymbol{D}:\boldsymbol{S}\}\boldsymbol{S} - \frac{1}{\lambda_{0\mathrm{b}}\Lambda^2}\left[3\alpha\Lambda^4\boldsymbol{S}^2 + \left(1-\alpha-3\alpha\Lambda^4\operatorname{tr}\left(\boldsymbol{S}^2\right)\right)\boldsymbol{S} - \frac{1-\alpha}{3}\boldsymbol{I}\right], \tag{2.37}$$

where $\lambda_{0\mathrm{b}}$ is the relaxation time of the backbone orientation (due to reptation), and α is the mobility factor characterizing the amount of anisotropic effect, similarly as in the Giesekus model. The backbone stress is given by:

$$\Lambda = \sqrt{1+\frac{\operatorname{tr}\left(\boldsymbol{\sigma}_e\right)}{3G_0}}. \tag{2.38}$$

However, an evolution equation for Λ has been derived within the XPP model:

$$\frac{\mathrm{D}\Lambda}{\mathrm{D}t} = \Lambda\{\boldsymbol{D}:\boldsymbol{S}\} - \frac{1}{\lambda_{\mathrm{s}}}(\Lambda-1), \tag{2.39}$$

where $\mathrm{D}/\mathrm{D}t$ is the material derivative,

$$\lambda_{\mathrm{s}} = \lambda_{0\mathrm{s}}\exp\left(-2(\Lambda-1)/q\right), \tag{2.40}$$

λ_{0s} is the stretch relaxation time, and q is the number of arms of the branched polymers.

We now derive a single evolution equation for the XPP model, in term of the conformation tensor defined as:

$$\boldsymbol{c} = 3\Lambda^2 \boldsymbol{S}. \tag{2.41}$$

Note that our definition of the conformation tensor differs with a factor 3 from the definition of McLeish and Larson (1998), in order to be consistent with equation (2.21). The change of variable yields the following expression for the upper convective derivative of the conformation tensor:

$$\begin{aligned}\overset{\nabla}{\boldsymbol{c}} &= \frac{\mathrm{D}}{\mathrm{D}t}\left(3\Lambda^2\boldsymbol{S}\right) - \left(3\Lambda^2\boldsymbol{S}\cdot\nabla\boldsymbol{u}^{\mathrm{T}} + \nabla\boldsymbol{u}\cdot 3\Lambda^2\boldsymbol{S}\right)\\ &= 6\boldsymbol{S}\frac{\mathrm{D}\Lambda}{\mathrm{D}t} + 3\Lambda^2\frac{\mathrm{D}\boldsymbol{S}}{\mathrm{D}t} - 3\Lambda^2\left(\boldsymbol{S}\cdot\nabla\boldsymbol{u}^{\mathrm{T}} + \nabla\boldsymbol{u}\cdot\boldsymbol{S}\right)\\ &= 6\boldsymbol{S}\Lambda\frac{\mathrm{D}\Lambda}{\mathrm{D}t} + 3\Lambda^2\overset{\nabla}{\boldsymbol{S}}.\end{aligned} \tag{2.42}$$

Incorporating equation (2.37) into equation (2.42) gives:

$$\overset{\nabla}{\boldsymbol{c}} = 6\boldsymbol{S}\Lambda\frac{\mathrm{D}\Lambda}{\mathrm{D}t} - 6\Lambda^2\left\{\boldsymbol{D}:\boldsymbol{S}\right\}\boldsymbol{S} - \frac{3}{\lambda_{0\mathrm{b}}}\left[3\alpha\Lambda^4\boldsymbol{S}^2 + \left(1-\alpha-3\alpha\Lambda^4\,\mathrm{tr}\left(\boldsymbol{S}^2\right)\right)\boldsymbol{S} - \frac{1-\alpha}{3}\boldsymbol{I}\right]. \tag{2.43}$$

Moreover, the first two terms at the right hand side can be substituted by $-6\Lambda\left(\Lambda-1\right)\boldsymbol{S}/\lambda_{\mathrm{s}}$, in virtue of equation (2.39), which further yields:

$$\overset{\nabla}{\boldsymbol{c}} = -\frac{1}{\lambda_{\mathrm{s}}}6\Lambda\left(\Lambda-1\right)\boldsymbol{S} - \frac{1}{\lambda_{0\mathrm{b}}}\left[\alpha\left(9\Lambda^4\boldsymbol{S}^2\right) + 3\left(1-\alpha-3\alpha\Lambda^4\,\mathrm{tr}\left(\boldsymbol{S}^2\right)\right)\boldsymbol{S} - \left(1-\alpha\right)\boldsymbol{I}\right]. \tag{2.44}$$

Thus, the change of variable (2.41) gives:

$$\overset{\nabla}{\boldsymbol{c}} = -\frac{1}{\lambda_{\mathrm{s}}}\frac{2\boldsymbol{c}}{\Lambda}\left(\Lambda-1\right) - \frac{1}{\lambda_{0\mathrm{b}}}\left[\alpha\boldsymbol{c}^2 + 3\left(1-\alpha-\alpha\frac{1}{3}\mathrm{tr}\left(\boldsymbol{c}^2\right)\right)\frac{\boldsymbol{c}}{3\Lambda^2} - \left(1-\alpha\right)\boldsymbol{I}\right]. \tag{2.45}$$

The rearranging of equation (2.45), and the use of equation (2.40) finally gives:

$$\overset{\nabla}{\boldsymbol{c}} = -\frac{1}{\lambda_{0\mathrm{b}}}\left(\alpha\boldsymbol{c}^2 + \frac{1}{\Lambda^2}\left[1-\left(1+\frac{1}{3}\mathrm{tr}\left(\boldsymbol{c}^2\right)\right)\alpha + \varpi\left(\Lambda^2-\Lambda\right)e^{2(\Lambda-1)/q}\right]\boldsymbol{c} - \left(1-\alpha\right)\boldsymbol{I}\right), \tag{2.46}$$

where the ratio $\varpi = 2\lambda_{0\mathrm{b}}/\lambda_{0\mathrm{s}}$ is also a material parameter characterizing the ratio of the reptation to the retraction time-scales. Finally, the change of variable (2.41) yield the following expression for the backbone stretch (2.38):

$$\Lambda = \sqrt{\mathrm{tr}\left(\boldsymbol{c}\right)/3}. \tag{2.47}$$

Note that $\mathrm{tr}\left(\boldsymbol{c}^2\right) = I_{\boldsymbol{c}}{}^2 - 2II_{\boldsymbol{c}}$ is also an invariant of the conformation tensor. Therefore, the strain and relaxation function of the XPP model are:

$$\begin{cases} f_{\mathrm{S}}\left(\boldsymbol{c}\right) = \boldsymbol{c} - \boldsymbol{I}, \\ f_{\mathrm{R}}\left(\boldsymbol{c}\right) = \alpha\boldsymbol{c}^2 + \dfrac{1}{\Lambda^2}\left[1-\left(1+\dfrac{1}{3}\mathrm{tr}\left(\boldsymbol{c}^2\right)\right)\alpha + \varpi\left(\Lambda^2-\Lambda\right)e^{2(\Lambda-1)/q}\right]\boldsymbol{c} - \left(1-\alpha\right)\boldsymbol{I}, \\ \quad \text{where} \quad \Lambda = \sqrt{\mathrm{tr}\left(\boldsymbol{c}\right)/3}. \end{cases} \tag{2.48}$$

2.2.2. The high Weissenberg number problem

The numerical simulations of viscoelastic flows are particularly prone to *numerical instabilities* due to the stiff hyperbolic nature of the constitutive equations. The *high Weissenberg number problem* (HWNP) has been one of the main challenges of computational rheology for more than two decades (Keunings 1986; Keunings 2000; Walters & Webster 2003). The HWNP manifests itself by a loss of reliability of the simulations, leading to a blowup of the numerical values. It is triggered when the Weissenberg number (or the Deborah number) reaches a critical value, which depends on the flow problem, the spatial discretization, and the numerical algorithm. Common stabilization strategies have consisted in enhancing the elliptic operator related to the viscous stresses, as for instance with the both-side-diffusion. However, this stabilization approach degrades the accuracy of transient solutions scheme (Xue, Tanner & Phan-Thien 2004), and values of magnitude ~ 1 are still considered as high values of the Weissenberg numbers. Kupferman (2005) has subsequently showed that the smoothness of the numerical solution can be destroyed by divergence-free stress perturbations, which do not couple back to the velocity field. Therefore, it was rather suggested to add stress diffusion terms into the constitutive models, in order to change the equation type from hyperbolic to parabolic (El-Kareh & Leal 1989), and ultimately to improve the numerical stability. Moreover, a small amount of isotropic stress diffusion has a physical justification, because of the Brownian motion of the polymer molecules. Although excessive artificial stress diffusion negatively affects the accuracy of numerical simulations (Min, Yoo & Choi 2001), small values do not alter the flow dynamics while enhancing numerical stability by maintaining a smooth and bounded stress solution (Sureshkumar & Beris 1995; Thomases 2011).

Lately, the origin of the HWNP was identified. First, it was found (Lee et al. 2004; Fattal & Kupferman 2004) that the loss of positive-definiteness of the *conformation tensor*—which should be symmetric positive-definite by definition—is a precursor of the HWNP. Secondly, it was shown that numerical instabilities arise from the under-resolution of spatial stress profiles (Fattal & Kupferman 2005; Hulsen, Fattal & Kupferman 2005), which is problematic, since the solutions of viscoelastic flows tend to have stress boundary layers with large variations in the stress gradients and exponential stress profiles near geometrical singularities (Renardy 2000). Failure to represent the stress gradients accurately (for instance due to polynomial interpolations of exponential profiles) results in underestimations of the convective stress fluxes, which are compensated by a multiplication of the stress growth rate, eventually leading to a numerical blowup.

Based on these observations, several techniques have been developed in order to enforce the positive-definiteness of the conformation tensor by construction. Today, the most popular method is the log–conformation representation of Fattal and Kupferman (2004; 2005), which consists in a change of variable using the matrix-logarithm of the conformation tensor. In addition to preserving the positive-definiteness, the log–conformation representation improves the representation of large stress gradients, since it linearizes the exponential stress profiles. The log–conformation representation has been implemented in various finite–volume and finite–element codes. Alternatively, Vaithianathan and Collins (2003) have presented two other schemes that guaranty the positive-definiteness of the conformation tensor, based on matrix decompositions. The first scheme uses the eigendecomposition, and evolves the eigenvalues and eigenvectors of the conformation tensor; the second scheme uses the Cholesky decomposition. Lozinski and Owens (2003) and Balci et al. (2011) have both proposed the square-root conformation representation, based on the matrix decomposition of the conformation tensor with its principal square-root (i.e. its unique positive-definite square-root). Other transformations preserving the positive-definiteness of the conformation have been discussed in Afonso, Pinho and Alves (2012). However, among all these different techniques, the log–conformation representation is the only one that linearizes the exponential stress profile, while for instance the square-root conformation representation only reduces the exponential growth (Chen et al. 2013). Thus, we chose to use the log–conformation representation because it has superiority in representing stress profiles with exponential variations.

2.2.3. The log–conformation representation

In the present work, the HWNP is remedied with the log–conformation representation of Fattal and Kupferman (2004; 2005). Note that our generic viscoelastic model (2.19),(2.21)–(2.23) is more general than the constitutive equations originally considered by Fattal and Kupferman (2004), but it is still compatible with the log–conformation representation. The benefit of our generic constitutive model is that it includes more viscoelastic models than the framework of Fattal and Kupferman (2004), notably the FENE–P, the Rolie–Poly and the XPP models, which are also very popular constitutive models derived from molecular theories.

The log–conformation representation for the two-dimensional case is summarized below. It consists in a change of variable in the constitutive model, in term of the matrix-logarithm of the conformation tensor:

$$\boldsymbol{\Psi} = \log \boldsymbol{c} . \tag{2.49}$$

Fattal and Kupferman (2004) have proven that the velocity gradient $\nabla \boldsymbol{u}$ can be decomposed as

$$\nabla \boldsymbol{u} = \boldsymbol{\Omega} + \boldsymbol{E} + \boldsymbol{N}\boldsymbol{c}^{-1} , \tag{2.50}$$

where $\boldsymbol{\Omega}$ and $\boldsymbol{N}$ are pure rotation (anti-symmetric) matrices, and $\boldsymbol{E}$ is a pure extension (symmetric traceless) matrix. When inserting the relations (2.49) and (2.50) into equation (2.22), the term $\boldsymbol{N}\boldsymbol{c}^{-1}$ vanishes, and, the constitutive equation (2.21) finally yields the following evolution equation for the log–conformation tensor:

$$\frac{\partial \boldsymbol{\Psi}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{\Psi} - (\boldsymbol{\Omega}\boldsymbol{\Psi} - \boldsymbol{\Psi}\boldsymbol{\Omega}) - 2\boldsymbol{E} = -\frac{1}{\lambda}\exp(-\boldsymbol{\Psi}) f_{\mathrm{R}}\left[\exp(\boldsymbol{\Psi})\right] . \tag{2.51}$$

In practice, the matrices $\boldsymbol{E}$ and $\boldsymbol{\Omega}$ are obtained from the projection of the velocity gradient into the principal base of the stress tensor. Let the eigendecomposition of the conformation tensor in a two-dimensional flow be

$$\boldsymbol{c} = \boldsymbol{R}\begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}\boldsymbol{R}^{\mathrm{T}} , \tag{2.52}$$

where λ_1 and λ_2 are the eigenvalues and $\boldsymbol{R}$ the orthogonal matrix containing the eigenvectors. Then the change of base of the velocity gradient is

$$\boldsymbol{R}^{\mathrm{T}}(\nabla \boldsymbol{u})\boldsymbol{R} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} . \tag{2.53}$$

Finally, the pure extension and rotation matrices in the reference base are obtained as:

$$\boldsymbol{E} = \boldsymbol{R}\begin{pmatrix} m_{11} & 0 \\ 0 & m_{22} \end{pmatrix}\boldsymbol{R}^{\mathrm{T}} , \tag{2.54}$$

and

$$\boldsymbol{\Omega} = \boldsymbol{R}\begin{pmatrix} 0 & \zeta \\ -\zeta & 0 \end{pmatrix}\boldsymbol{R}^{\mathrm{T}} , \tag{2.55}$$

where

$$\zeta = \frac{m_{12} + m_{21}}{\lambda_2 - \lambda_1} . \tag{2.56}$$

In the special case where the reference base is the principal base of the stress tensor, i.e. $\boldsymbol{c}$ is already diagonal, then $\boldsymbol{E} = \boldsymbol{D}$ and $\boldsymbol{\Omega} = \boldsymbol{0}$.

In summary, the viscoelastic stress solver finds the solution of equation (2.51) and recovers the conformation tensor with the matrix-exponential of $\boldsymbol{\Psi}$:

$$\boldsymbol{c} = \exp(\boldsymbol{\Psi}) . \tag{2.57}$$

The formula (2.57) enforces by construction the symmetric positive-definiteness of the conformation tensor. Note that the matrix-logarithm and the matrix-exponential are tensor operations, where the logarithm and the exponent are applied to the eigenvalues of a tensor, for instance:

$$\log(\boldsymbol{c}) \equiv \boldsymbol{R}\begin{pmatrix} \log\lambda_1 & 0 \\ 0 & \log\lambda_2 \end{pmatrix}\boldsymbol{R}^T. \tag{2.58}$$

2.3. References

Afonso, AM, Pinho, FT & Alves, MA 2012, 'The kernel-conformation constitutive laws', *Journal of Non-Newtonian Fluid Mechanics*, vol. 167, pp. 30–37.

Balci, N, Thomases, B, Renardy, M & Doering, CR 2011, 'Symmetric factorization of the conformation tensor in viscoelastic fluid models', *Journal of Non-Newtonian Fluid Mechanics*, vol. 166, pp. 546–553.

Bird, RB, Armstrong, RC, Hassager, O 1987, *Dynamics of polymeric liquids. Volume 1: fluid mechanics*, 2nd eds, A Wiley-Interscience Publication, John Wiley & Sons.

Bird, RB, Dotson, PJ & Johnson, NL 1980, 'Polymer solution rheology based on a finitely extensible bead—spring chain model', *Journal of Non-Newtonian Fluid Mechanics*, vol. 7, pp. 213–235.

Bird, RB & Wiest, JM 1985, 'Anisotropic effects in dumbbell kinetic theory', *Journal of Rheology*, vol. 29, pp. 519–532.

Chen, X, Marschall, H, Schäfer, M & Bothe, D 2013, 'A comparison of stabilisation approaches for finite-volume simulation of viscoelastic fluid flow', *International Journal of Computational Fluid Dynamics*, vol. 27, pp. 229–250.

Chilcott, MD & Rallison, JM 1988, 'Creeping flow of dilute polymer solutions past cylinders and spheres', *Journal of Non-Newtonian Fluid Mechanics*, vol. 29, pp. 381–432.

El-Kareh, AW & Leal, LG 1989, 'Existence of solutions for all Deborah numbers for a non-Newtonian model modified to include diffusion', *Journal of Non-Newtonian Fluid Mechanics*, vol. 33, pp. 257–287.

Fattal, R & Kupferman, R 2004, 'Constitutive laws for the matrix-logarithm of the conformation tensor', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 281–285.

Fattal, R & Kupferman, R 2005, 'Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 126, pp. 23–37.

Frigaard, IA & Nouar, C 2005, 'On the usage of viscosity regularisation methods for visco-plastic fluid flow computation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 127, pp. 1–26.

Giesekus, H 1982, 'A simple constitutive equation for polymer fluids based on the concept of deformation-dependent tensorial mobility', *Journal of Non-Newtonian Fluid Mechanics*, vol. 11, pp. 69–109.

Hulsen, MA, Fattal, R & Kupferman, R 2005, 'Flow of viscoelastic fluids past a cylinder at high Weissenberg number: stabilized simulations using matrix logarithms', *Journal of Non-Newtonian Fluid Mechanics*, vol. 127, pp. 27–39.

Keunings, R 1986, 'On the high Weissenberg number problem', *Journal of Non-Newtonian Fluid Mechanics*, vol. 20, pp. 209–226.

Keunings, R 2000, 'A survey of computational rheology', in: *Proceedings of the XIIIth International Congress on Rheology*, pp. 7–14, The British Society of Rheology, Cambridge.

Kramers, HA 1946, 'The behavior of macromolecules in inhomogeneous flow', *The Journal of Chemical Physics*, vol. 14, pp. 415–424.

Kupferman, R 2005, 'On the linear stability of plane Couette flow for an Oldroyd-B fluid and its numerical approximation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 127, pp. 169–190.

Kwon, Y & Leonov, AI 1995, 'Stability constraints in the formulation of viscoelastic constitutive equations', *Journal of Non-Newtonian Fluid Mechanics*, vol. 58, pp. 25–46.

Lee, J, Yoon, S, Kwon, Y & Kim, S 2004, 'Practical comparison of differential viscoelastic constitutive equations in finite element analysis of planar 4:1 contraction flow', *Rheologica Acta*, vol. 44, pp. 188–197.

Likhtman, AE & Graham, RS 2003, 'Simple constitutive equation for linear polymer melts derived from molecular theory: Rolie–Poly equation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 114, pp. 1–12.

Lozinski, A & Owens, RG 2003, 'An energy estimate for the Oldroyd B model: theory and applications', *Journal of Non-Newtonian Fluid Mechanics*, vol. 112, pp. 161–176.

McLeish, TCB & Larson, RG 1998, 'Molecular constitutive equations for a class of branched polymers: The pom-pom polymer', *Journal of Rheology*, vol. 42, pp. 81–110.

McLeish, TCB 2002, 'Tube theory of entangled polymer dynamics', *Advances in Physics*, vol. 51, pp. 1379–1527.

Min, T, Yoo, JY & Choi H 2001, 'Effect of spatial discretization schemes on numerical solutions of viscoelastic fluid flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 100, pp. 27–47.

Mitsoulis, E 2007, 'Flows of viscoplastic materials: models and computations', *Rheology reviews*, vol. 5, pp. 135–178.

Papanastasiou, TC 1987, 'Flows of materials with yield', *Journal of Rheology*, vol. 31, pp. 385–404.

Peterlin, A 1966, 'Hydrodynamics of macromolecules in a velocity field with longitudinal gradient', *Journal of Polymer Science Part B: Polymer Letters*, vol. 4, pp. 287–291.

Phan-Thien, N & Tanner, RI 1977, 'A new constitutive equation derived from network theory', *Journal of Non-Newtonian Fluid Mechanics*, vol. 2, pp. 353–365.

Phan-Thien, N 1978, 'A nonlinear network viscoelastic model', *Journal of Rheology*, vol. 22, pp. 259–283.

Oldroyd, JG 1950, 'On the formulation of rheological equations of state', *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 200, pp. 523–541.

Renardy, M 2000, 'Current issues in non-Newtonian flows: a mathematical perspective', *Journal of Non-Newtonian Fluid Mechanics*, vol. 90, pp. 243–259.

Sureshkumar, R & Beris, AN 1995, 'Effect of artificial stress diffusivity on the stability of numerical calculations and the flow dynamics of time-dependent viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 60, pp. 53–80.

Thomases, B 2011, 'An analysis of the effect of stress diffusion on the dynamics of creeping viscoelastic flow', *Journal of Non-Newtonian Fluid Mechanics*, vol. 166, pp. 1221–1228.

Vaithianathan, T & Collins, LR 2003, 'Numerical approach to simulating turbulent flow of a viscoelastic polymer solution', *Journal of Computational Physics*, vol. 187, pp. 1–21.

Verbeeten, WM, Peters, GW & Baaijens, FP 2001, 'Differential constitutive equations for polymer melts: the eXtended Pom–Pom model', *Journal of Rheology*, vol. 45, pp. 823–843.

Xue, SC, Tanner, RI & Phan-Thien, N 2004, 'Numerical modelling of transient viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 33–58.

Walters, K & Webster, MF 2003, 'The distinctive CFD challenges of computational rheology', *International Journal for Numerical Methods in Fluids*, vol. 43, pp. 577–596.

Watanabe, H 1999, 'Viscoelasticity and dynamics of entangled polymers', *Progress in Polymer Science*, vol. 24, pp. 1253–1403.

Warner Jr, HR 1972, 'Kinetic theory and rheology of dilute suspensions of finitely extendible dumbbells', *Industrial & Engineering Chemistry Fundamentals*, vol. 11, pp. 379–387.

Chapter 3

Conservation laws

Alongside the calculation of the viscoelastic extra-stress from the constitutive model, the conservation laws (i.e. the equations of motion) are solved in order to update the kinematics of the flow.

This chapter briefly presents the conservation laws and reviews the various approaches that are used in computational fluid dynamics to solve the conservation of mass and momentum. Advantages and drawbacks of the different methods are discussed in the context of a viscoelastic creeping flow. Also presented in this chapter is the pressureless *pure streamfunction formulation* that has been used to generate the numerical results presented in chapter 6.

3.1. Velocity–pressure formulation

The conservation laws include the mass conservation—i.e. the continuity equation for incompressible fluids—and the conservation of the linear momentum:

$$\nabla \cdot \boldsymbol{u} = 0 \tag{3.1}$$

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}\right) = -\nabla p + \nabla \cdot \boldsymbol{\tau} + \boldsymbol{f} \tag{3.2}$$

where ρ is the density of the fluid, ∇p is the pressure gradient, $\nabla \cdot \boldsymbol{\tau}$ is the divergence of the internal stresses build-up in the non–Newtonian fluid (calculated from the constitutive model), and $\boldsymbol{f}$ is a vector of the external forces (for instance the gravity). In these equations, the conservation laws are formulated in terms of velocity and pressure as the primitive variables.

The energy is also a conserved physical quantity during thermo-dynamical and thermo-mechanical processes. The energy conservation is typically solved in order to compute the variations in the temperature of the physical system. In such models, the pressure variable can be linked to the temperature via an equation of state. Nevertheless, only the isothermal case has been considered in this work; thus the energy equation is not solved. Consequently, the pressure variable is not a thermodynamical variable. The pressure gradient can rather be seen as a Lagrange multiplier of the incompressibility constraint in the momentum equation (Perot 1993).

The momentum equation (3.2) and the constitutive equation expressed in terms of the logarithmic–conformation tensor (see chapter 2) can easily be solved sequentially, for instance with successive substitution iterations, as they both have time-derivative terms. Concerning the coupling of the velocity with the pressure, the situation is different because there is no time-derivative in the equation for the pressure, which is either the

continuity equation (3.1) or the pressure Poisson equation obtained by taking the divergence of (3.2) and invoking the incompressibility:

$$\nabla^2 p = \nabla \cdot \left(\nabla \cdot \boldsymbol{\tau} + \boldsymbol{f} - \rho \boldsymbol{u} \cdot \nabla \boldsymbol{u} \right). \tag{3.3}$$

Therefore, the decoupling of the velocity and pressure is more difficult than the decoupling of the velocity and the logarithmic–conformation tensor.

The discretization of the system of equations (3.1)–(3.2) yields the following matrix system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_\mathbf{u} \\ \mathbf{0} \end{bmatrix}, \tag{3.4}$$

where $\mathbf{u}^{n+1}$ and $\mathbf{p}$ are the new solution vectors of the discretized velocity and pressure variables. The superscript *n+1* refers to the solution at the end of the time step. No superscript is placed on the pressure variables because it is a Lagrange multiplier. Thus, it corresponds to the average of the pressure over the entire time interval. However, if $\mathbf{S}_\mathbf{u}$ contains explicit pressure gradient terms from the previous time step, then $\mathbf{p}$ is interpreted as a pressure increment $\delta\mathbf{p} = \mathbf{p}^{n+1} - \mathbf{p}^n$ and the discretization is said to be *incremental*. The matrix $\mathbf{A}$ is the Jacobi matrix of the velocities in the momentum equations, $\mathbf{G}$ is the matrix accounting for the gradients in the momentum equations, and $\mathbf{D}$ is the matrix accounting for the divergence of the velocity in the continuity equations. The vector $\mathbf{S}_\mathbf{u}$ contains the sources terms of the momentum equations (for instance the divergence of the viscoelastic stress), the explicit contributions of the velocity, and the boundary conditions. The term $\mathbf{0}$ is the null matrix or the null vector. The three matrices $\mathbf{A}$, $\mathbf{G}$ and $\mathbf{D}$ depend on the spatial discretization. The matrix $\mathbf{A}$ and the vector $\mathbf{S}_\mathbf{u}$ additionally depend on the time discretization. The inversion of the monolithic matrix $\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix}$ all at once would lead to the discrete solution $\begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p} \end{bmatrix}$ of the conservation laws. However, the monolithic matrix is severely ill-conditioned because of the zero–block–diagonal matrix that comes from the absence of the pressure terms in the continuity equations.

3.1.1. Pseudo–compressibility methods

The pseudo–compressibility methods remedy the ill-conditioned of the monolithic matrix system by the introduction of the missing pressure terms in the continuity equation. Thus, as their name suggests, the pseudo–compressible methods relax the incompressibility constraint. Various classes of pseudo–compressibility methods exist, depending on the additional pressure term that is used. In the *penalty method* (Hughes, Liu & Brooks 1979; Shen 1995), the additional term is simply proportional to the pressure field:

$$\nabla \cdot \boldsymbol{u} + \varepsilon p = 0, \tag{3.5}$$

where ε is the pseudo-compressibility parameter. In the *artificial compressibility method* (Chorin 1967; Qian, Zhang & Li 2010), the continuity equation contains a time-derivative term of the pressure:

$$\nabla \cdot \boldsymbol{u} + \varepsilon \frac{\partial p}{\partial t} = 0. \tag{3.6}$$

The *pressure–stabilization method* (Brezzi & Douglas Jr 1988; Shen 1992) uses the Laplacian of the pressure field:

$$\nabla \cdot \boldsymbol{u} - \varepsilon \Delta p = 0. \tag{3.7}$$

More recently, a new *pseudo–compressibility method* (Shen 1996; Shen 1997) was introduced, using the time derivative of the Laplacian of the pressure:

$$\nabla \cdot \boldsymbol{u} - \varepsilon \frac{\partial}{\partial t}(\Delta p) = 0 . \tag{3.8}$$

The pseudo–compressibility methods results in a regularized modified matrix system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{Q} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{\mathbf{u}} \\ \mathbf{0} \end{bmatrix}, \tag{3.9}$$

where $\mathbf{Q}$ is the matrix operator accounting for the additional pressure term in the continuity equation. The regularized coupled matrix system (3.9) can be solved all-at-once (without segregation steps). However, the solution of the regularized matrix system (3.9) is only an approximate solution of the incompressible flow governed by equations (3.1)–(3.2).

The conditioning number of the matrix system (3.9) depends on the pseudo-compressibility parameter ε. On the one hand, large values of ε improve the conditioning, but it relaxes the incompressible constraint. On the other hand, small values of the parameter ε increase the fidelity to the incompressible fluid, however it results in a severely ill-conditioned matrix system when $\varepsilon \to 0$. Thus, the choice of the perturbation parameter ε is crucial for the accuracy of the method, and there is a trade-off between numerical precision and physical accuracy. Moreover, the optimum value for ε depends on the flow problem and its discretization.

In the case of a viscoelastic fluid, it is preferable to avoid pseudo–compressibility methods because it is suspected that the combination of compressibility and elasticity could produce self-sustained spurious oscillations, since both have the ability to store energy. Potential numerical artefacts could include compressive waves, elastic waves and other artificial instabilities, interfering with the true solution of the incompressible viscoelastic flow, which may also contain physical instabilities.

3.1.2. Velocity–pressure segregation methods

The *velocity–pressure segregation methods* address the issue of the ill-conditioning problem of the matrix system (3.4) by dividing the coupled velocity–pressure formulation into two sub-problems, in order to solve sequentially the velocity and pressure. Hence, these methods are very attractive; however the decoupling introduces splitting errors. In all these methods, tentative velocities and pressures are computed from the momentum equation and a Poisson-like pressure equation, respectively. Additional velocity and/or pressure corrections are calculated from the tentative solution. The various velocity–pressure segregation methods can be classified into different families, although efforts have been made to unify the different approaches into a general framework (Ni & Abdou 2007). On the one hand, the semi-implicit iterative methods (also referred to as the *pressure–correction methods*), such as SIMPLER, SIMPLEC or PISO (Patankar 1980; Issa 1986; Versteeg & Malalasekera 2007), solve the conservation laws (3.1)–(3.2) with a semi-implicit scheme, and successive iterations between the pressure and velocity corrections. However, these methods solve the conservation of mass and momentum to a certain tolerance and an accurate solution can require a lot of iterations. On the other hand, the decoupling in the non-interactive methods (also referred to as *projection methods* or *fractional step methods*), initially presented as a projection of the tentative velocity onto the sub-space of divergence-free vector fields (Chorin 1968; Kim & Moin 1985; Bell, Colella & Glaz 1989), can also be seen as an algebraic splitting or an approximate factorization of the monolithic matrix $\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix}$ (Dukowicz & Dvinsky 1992; Perot 1993).

The exact LU block matrix decomposition gives:

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.10}$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{A}^{-1}$ is the inverse of the $\mathbf{A}$. Thus, the matrix system

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{\mathbf{u}} \\ \mathbf{0} \end{bmatrix} \tag{3.11}$$

can be split as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{*} \\ \mathbf{p}^{*} \end{bmatrix} = \begin{bmatrix} \mathbf{S}_{\mathbf{u}} \\ \mathbf{0} \end{bmatrix}, \tag{3.12}$$

$$\begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n+1} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{u}^{*} \\ \mathbf{p}^{*} \end{bmatrix}, \tag{3.13}$$

which yields the following exact fractional steps:

$$\begin{cases} \mathbf{A}\mathbf{u}^{*} = \mathbf{S}_{\mathbf{u}} \\ \mathbf{D}\mathbf{A}^{-1}\mathbf{G}\mathbf{p} = \mathbf{D}\mathbf{u}^{*} \\ \mathbf{u}^{n+1} = \mathbf{u}^{*} - \mathbf{A}^{-1}\mathbf{G}\mathbf{p} \end{cases} \tag{3.14}$$

where $\mathbf{u}^{*}$ is an intermediate velocity. At this stage, the algebraic splitting (3.10) provides the exact solution of the matrix system (3.4). However, the inverse $\mathbf{A}^{-1}$ of the Jacobi matrix is an unknown which is very expensive to compute directly. Hence, the fractional step methods utilize approximate inverses $\mathbf{B} \approx \mathbf{A}^{-1}$. The different fractional-step methods vary in their approximations of $\mathbf{B}$. Therefore, the fractional-step methods produce errors in the velocity and the pressure solution. The incompressibility constraint is enforced if the fractional-step methods use the same approximate inverses $\mathbf{B}$ in the second and last stages in (3.14). However, since the continuity equation is solved indirectly, the mass conservation is usually not fulfilled to the machine precision, but its accuracy depends on the conditioning number of the matrix product $\mathbf{BC}$.

In the typical discretization of the momentum equation (3.2), the viscous stresses are treated implicitly (for a stability reason) and the inertial terms are linearized, resulting in the following expression for the matrix $\mathbf{A}$:

$$\mathbf{A} = \frac{1}{\Delta t}\left[\mathbf{I} + \Delta t\left(\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right)\right], \tag{3.15}$$

where $\mathbf{L}$ is the Laplace operator for of velocity (representing the viscous stress, or momentum diffusion), $\mathbf{N}$ is the linearized operator of the inertial terms, Δt is the time-increment of the discretization, Re is the Reynolds number (a dimensionless quantity expressing the ratio of inertial forces to viscous forces), and $\beta \in [0,1]$ is the solvent viscosity ratio ($\beta = 1$ for inelastic fluid; see chapter 2). Note that $\mathbf{N}$ is absent in (3.15) if the inertial terms are treated explicitly; however, in the case of flows at low Re, the treatments of the inertial terms is unimportant since they are negligible. Following the analysis of Perot (1993), approximate inverses of $\mathbf{A}$ can be constructed with the following series expansion:

$$\mathbf{A}^{-1} = \Delta t\sum_{k=0}^{\infty}\left[-\Delta t\left(\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right)\right]^{k}, \tag{3.16}$$

which converges when (Henriksen & Holmen 2002):

$$\Delta t < \left\|\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right\|_{2}^{-1}. \tag{3.17}$$

Thus, approximate inverses of arbitrary order of accuracy *N* with respect to Δt can be constructed as:

$$\mathbf{B} = \Delta t\sum_{k=0}^{N}\left[-\Delta t\left(\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right)\right]^{k} + O\left(\Delta t^{N+1}\right). \tag{3.18}$$

The decoupling numerical error introduced by a fractional-step method is:

$$e = \left\| \mathbf{AB} - \mathbf{I} \right\|. \tag{3.19}$$

This error adds up to the other sources of numerical errors (discretization, iterative solver, round-off, etc.) and it translates into non-conservation of the momentum. The decoupling errors are generally considered acceptable if the approximate inverse $\mathbf{B}$ has an order of accuracy *N* equal or higher than the order of accuracy of the discretization. However, a dimensional analysis of the decoupling errors shows that:

$$e \propto \Delta t \left| \frac{\beta}{Re} - 1 \right|^N. \tag{3.20}$$

Consequently, the decoupling errors become large at low Re numbers (and especially when $Re < \beta/2$), unless Δt gets very small. This can also be explained intuitively: at low Re, the flow is driven by the pressure gradients (since the inertia effects are negligible); therefore the coupling between viscous stress and the pressure gradient is strong. Furthermore, the relation in (3.20) shows that beyond a critical case where $Re < \beta/2$, the decoupling errors become very large, and get worse with higher order *N* of the decoupling method. In this case, limiting the decoupling errors by low time-step sizes would become immensely onerous. Therefore, the decoupled velocity–pressure approach used in this fraction-step method becomes inefficient and inaccurate at a low *Re* number.

Other variants of fractional-step methods with different choices of the approximate inverse $\mathbf{B}$ are reviewed by Quarteroni, Saleri and Veneziani (2000), and Guermond, Minev and Shen (2006). A recent study of Sousa, Oishi and Buscaglia (2015), on the performance of the projection methods for low Re simulations, reaches the same conclusion. In particular it is shown that non-incremental methods require very small time steps (of the same order of magnitude as for the explicit method), and that incremental methods develop spurious transients, unless small time steps are used.

3.1.3. Drawbacks of the velocity–pressure formulation in viscoelastic flows

In the velocity–pressure flow formulation, the isostatic pressure field does not have an evolution equation, but it is determined through the coupling of equations (3.1) and (3.2). The isostatic pressure is defined as the isotropic stress component of the material:

$$p \equiv -\frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma}), \tag{3.21}$$

where

$$\boldsymbol{\sigma} \equiv -p\boldsymbol{I} + \boldsymbol{s} \tag{3.22}$$

is the Cauchy stress, $\boldsymbol{I}$ is the identity matrix, and $\boldsymbol{s}$ is the deviatoric stress component. Since the entire Cauchy stress must contribute to the conservation of momentum, equations (3.2), (3.21) and (3.22) imply that

$$\boldsymbol{s} = \boldsymbol{\tau}. \tag{3.23}$$

For inelastic (purely viscous) materials, $\boldsymbol{\tau} = 2\eta\boldsymbol{D}$ is by definition traceless, hence the consistency between equations (3.21)–(3.23). However, in the case of viscoelastic materials, the material response decomposes into a purely viscous part $\boldsymbol{\tau}_\mathrm{S}$ (the solvent contribution), and a visco-elastic part $\boldsymbol{\tau}_\mathrm{P}$ (the polymer contribution), which also contributes to the isotropic stress. Thus, the Cauchy stress becomes:

$$\boldsymbol{\sigma} = -p\boldsymbol{I} + 2\eta\boldsymbol{D} + \boldsymbol{\tau}_\mathrm{P}, \tag{3.24}$$

which is inconsistent with the pressure definition (3.21), since the viscoelastic stress component is not traceless in general, i.e. $\mathrm{tr}(\boldsymbol{\tau}_\mathrm{P}) \neq 0$. Fortunately, the pressure field in an incompressible flow is not a thermodynamic state variable (as opposed to the case of compressible flows); therefore this inconsistency does not affect the

dynamics of the flow (Bollada & Phillips 2012). This could explain why this inconsistency has received relatively little attention in the computational rheology literature.

Oliveira (2000) has proposed a traceless viscoelastic stress formulation for the Oldroyd-B model, where an augmented pressure $\tilde{p}$ is defined as the true isotropic stress component

$$\tilde{p} \equiv p - \frac{1}{3}\mathrm{tr}\left(\boldsymbol{\tau}_{\mathrm{P}}\right), \tag{3.25}$$

and a new constitutive equation is derived for the deviatoric part of the viscoelastic stress

$$\tilde{\boldsymbol{\tau}}_{\mathrm{P}} \equiv \boldsymbol{\tau}_{\mathrm{P}} - \frac{1}{3}\mathrm{tr}\left(\boldsymbol{\tau}_{\mathrm{P}}\right)\boldsymbol{I}. \tag{3.26}$$

Although both formulations are equivalent from a dynamical point of view, the results of Oliveira shows that the traceless viscoelastic stress formulation improves the convergence of the numerical algorithm, i.e. it is numerically more stable. Oliveira tested the traceless viscoelastic stress formulation in two geometries, with and without a geometrical singularity, for Weissenberg numbers ranging from 0.3 to 1.5. In these numerical experiments, the traceless viscoelastic stress formulation saved between 24% and 84% of the total number of iterations, as compared to the standard pressure formulation. Moreover, the higher the Weissenberg number, the larger the gain in computational efficiency. This result suggests that the polymer contribution to the Cauchy stress degrades the iterative convergence of the velocity–pressure–extra-stress segregated solver algorithms. In general, an iterative segregated algorithm that is used to solve more than two coupled variables can be prone to convergence difficulties, if two of the variables have competing effects on the third. This is the case of the standard velocity–pressure–extra-stress decoupled approach, where both the pressure p and the isotropic part of the elastic stress $\mathrm{tr}\left(\boldsymbol{\tau}_{\mathrm{P}}\right)/3$ interact with the velocity field as Lagrange multipliers of the continuity constraint. It can be seen in the simulations of Kwon (2012) that fluctuations in the viscoelastic stress cause fluctuations in the pressure field. Therefore, it is reasonable to assume that numerical errors propagate to the different variables because of the segregated algorithm. Moreover, the absence of direct coupling in the equations between the pressure and viscoelastic stress, which could have stabilized the algorithm, negatively impacts the convergence of the decoupled velocity–pressure–extra-stress methods.

Xue, Tanner and Phan-Thien (2004) have investigated the efficiency of several algorithms belonging to the SIMPLE family, in the context of viscoelastic flows. In their numerical experiments, they considered the 1D start-up and decay of plane Poiseuille flow with an Oldroyd-B fluid, at $Re = 0.05$. They showed that above a threshold value of the Courant number, the transient solution of the viscoelastic flow is prone to unphysical oscillations, although all the governing equations are temporarily discretized with unconditionally stable schemes. These numerical instabilities come from the semi-implicit velocity–pressure decoupling. They have determined that the SIMPLEST and PISO schemes, which are the most accurate algorithms of the SIMPLE family, are also the most stable (i.e. they have higher threshold Courant number), in both the Newtonian and the viscoelastic cases. The threshold Courant number for stable calculations also depends on the temporal discretization scheme. For instance, the threshold values for the SIMPLEST algorithm were $C = 2$ for the Crank-Nicolson scheme, and $C = 3.3$ for the two-level backward differentiation formula. However, the test case that they considered is relatively simple: the 1D nature of the problem disables the coupling between some of the velocity and stress components, which are zero in this case. Moreover, the constant and uniform solution for the pressure gradient along the channel, which is entirely determined by the pressure difference imposed at the boundaries, produces a particularly mild coupling between the velocity and pressure. Therefore, lower threshold Courant numbers are to be expected in less favorable cases of complex flows.

In summary, the velocity–pressure segregation methods present several limitations in the solving of highly viscous and/or viscoelastic flow:

- the competing effects of the pressure and the extra-stress terms degrade the iterative convergence of segregated solvers,

- the decoupled velocity–pressure approach produces additional numerical errors that can trigger unphysical oscillations,
- the splitting technique requires restrictions of the time-step size, which become very severe at low Re,
- the discrete continuity is not fulfilled exactly, potentially producing artificial compressibility effects.

For all these reasons, the use of a pressureless flow formulation (i.e. based on the vorticity equation ensuing from the conservation of angular momentum) is justified, in order to improve the accuracy and the stability of the viscoelastic flow solvers.

3.2. Vorticity formulation

The inherent numerical difficulties due to decoupling of the velocity–pressure flow formulation can be avoided with a pressureless formulation based on the vorticity equation. In this section, the conservation laws are reformulated in term of the vorticity conservation (i.e. the conservation of the angular momentum); thus removing all the previously mentioned drawbacks of the decoupled velocity–pressure methods.

By means of the vector calculus identity $\nabla\times(\nabla p)=0$ for any scalar field p, the pressure unknown vanishes from the conservation laws by taking the curl of the momentum equation (3.2), yielding the following equation:

$$\frac{\partial \boldsymbol{\omega}}{\partial t}+\nabla\times\left((\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\right)=\nabla\times\left(\frac{\nabla\cdot\boldsymbol{\tau}}{\rho}\right), \tag{3.27}$$

where

$$\boldsymbol{\omega}=\nabla\times\boldsymbol{u} \tag{3.28}$$

is the vorticity of the flow. The curl of the inertia term (the second term of the left hand side) is further treated using the following vectorial identities:

$$(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}=\frac{1}{2}\nabla(\boldsymbol{u}\cdot\boldsymbol{u})-\boldsymbol{u}\times(\nabla\times\boldsymbol{u}) \tag{3.29}$$

and

$$\nabla\times(\boldsymbol{u}\times\boldsymbol{\omega})=\boldsymbol{u}(\nabla\cdot\boldsymbol{\omega})-\boldsymbol{\omega}(\nabla\cdot\boldsymbol{u})+(\boldsymbol{\omega}\cdot\nabla)\boldsymbol{u}-(\boldsymbol{u}\cdot\nabla)\boldsymbol{\omega}, \tag{3.30}$$

where $\nabla\cdot\boldsymbol{\omega}=0$ by definition, and $\nabla\cdot\boldsymbol{u}=0$ because of the incompressibility. In addition, the curl of the purely viscous stress becomes:

$$\nabla\times\left(\frac{\nabla\cdot 2\eta_S\boldsymbol{D}}{\rho}\right)=\frac{\beta\eta_0}{\rho}\nabla^2\boldsymbol{\omega}. \tag{3.31}$$

Finally combining equations (3.27)–(3.31) yields the vorticity equation:

$$\frac{\partial \boldsymbol{\omega}}{\partial t}+(\boldsymbol{u}\cdot\nabla)\boldsymbol{\omega}-(\boldsymbol{\omega}\cdot\nabla)\boldsymbol{u}=\frac{\eta_0}{\rho}\left[\beta\nabla^2\boldsymbol{\omega}+\frac{1-\beta}{\lambda}\nabla\times\left(\nabla\cdot f_S(\boldsymbol{c})\right)\right]. \tag{3.32}$$

The vorticity equation (3.32) is used to compute the new vorticity. The new velocity can be updated from the velocity Poisson equation, derived from the continuity:

$$\nabla^2\boldsymbol{u}=-\nabla\times\boldsymbol{\omega}. \tag{3.33}$$

The equations (3.32) and (3.33) are the bases of the velocity–vorticity formulation of the Navier–Stokes equations (Guj & Stella 1993; Meitz & Fasel 2000; Lo, Murugesan & Young 2005). One difficulty with the vorticity-based formulation is the correct determination of the vorticity generated at boundaries (Weinan & Liu 1996a), since there is no natural boundary condition for the vorticity. Moreover, the vorticity generated at the wall boundary is the central physical quantity for the flow dynamics, in many applications.

The vorticity–streamfunction flow formulation is another popular flow formulation based on the vorticity transport equation, where the velocity is linked to a streamfunction potential (Weinan & Liu 1996b; Weinan & Liu 1997; Shukla, Tatineni & Zhong 2007); more details are given in the following section.

3.3. Two-dimensional pure streamfunction formulation

The pure streamfunction formulation has been proposed by Kupferman (2001) as an alternative to the vorticity formulation. The incompressibility constraint is enforced using the *Helmholtz decomposition*, which states that any vector field that is twice continuously differentiable at all the points in its domain can be decomposed into the sum of an irrotational (curl-free) vector field and a solenoidal (divergence-free) vector field. Hence, there exits scalar and vector potentials, φ and $\boldsymbol{\Phi}$, respectively associated with the irrotational and solenoidal velocity components, such that:

$$\boldsymbol{u} = \nabla\varphi + \nabla\times\boldsymbol{\Phi}\,. \tag{3.34}$$

These potentials are not unique, but defined up to an arbitrary constant. For an incompressible flow, the velocity field is by definition solenoidal; hence its irrotational component $\nabla\varphi$ must be zero, yielding the following equation:

$$\boldsymbol{u} = \nabla\times\boldsymbol{\Phi}\,. \tag{3.35}$$

Combining relations (3.33) and (3.35) also yields the following constraint:

$$\boldsymbol{\omega} = -\nabla^2\boldsymbol{\Phi} \tag{3.36}$$

Thus, a unique dynamical equation can be derived by replacing all the velocity and vorticity unknowns in the vorticity transport equation (3.32) by the streamfunction unknown, using equations (3.35) and (3.36).

In the two-dimensional case, the velocity vector has two in-plane components, while the vorticity and the streamfunction fields only have their out-of-plane component which is non-zero:

$$\boldsymbol{u} = \left(u,v,0\right)^{\mathrm{T}},\ \boldsymbol{\omega} = \left(0,0,\omega\right)^{\mathrm{T}},\ \boldsymbol{\Phi} = \left(0,0,\phi\right)^{\mathrm{T}}. \tag{3.37}$$

Thus, the two-dimensional flow is governed by a single equation accounting for both the momentum conservation and the incompressibility constraint:

$$\frac{\partial\nabla^2\phi}{\partial t} + \frac{\partial\phi}{\partial y}\frac{\partial\nabla^2\phi}{\partial x} - \frac{\partial\phi}{\partial x}\frac{\partial\nabla^2\phi}{\partial y} = \frac{\eta_0}{\rho}\left[\beta\nabla^4\phi - \frac{1-\beta}{\lambda}\nabla\times\left(\nabla\cdot f_{\mathrm{S}}\left(\boldsymbol{c}\right)\right)\right], \tag{3.38}$$

where the term $\nabla^4\phi$ is the biharmonic operator of the streamfunction. Then, the two components of the velocity vector are recovered as:

$$\begin{aligned} u &= \partial\phi/\partial y\,,\\ v &= -\partial\phi/\partial x\,, \end{aligned} \tag{3.39}$$

which automatically fulfills the continuity constraint to the tolerance of the machine precision. Hence, the pure streamfunction flow formulation yields a fully implicit scheme, without splitting or fractional steps.

The solution of the streamfunction equation (3.38) requires boundary conditions for the streamfunction, as well as its derivatives, which are obtained from the boundary conditions of the velocity. The pressure and traction boundary conditions are not implementable as such, although tangential pressure gradients to the boundary, representing for instance a hydrostatic pressure, can be included as source terms in the streamfunction equation. Consequently, the boundary conditions for the flows which are driven by pressure differences must be specified in terms of the corresponding inflow velocity profiles and fully developed outflows. The discretization of the pure streamfunction formulation and the implementation of the boundary conditions are discussed in the next chapter 4.

Finally, the pressure field can be post-processed by solving the Poisson equation, obtained by taking the divergence of the momentum equations. By virtue of the continuity, the divergence of the velocity time-derivatives and the purely viscous stresses vanishes; hence yielding:

$$\nabla^2 p = \nabla \cdot \left(\nabla \cdot \boldsymbol{\tau}_{\mathrm{P}} + \boldsymbol{f} \right) - \rho \nabla \cdot \left(\boldsymbol{u} \cdot \nabla \boldsymbol{u} \right). \tag{3.40}$$

The Neumann pressure boundary conditions are obtained from the projection of the moment equation along the normal vector of the boundaries. In opposition to the velocity–pressure methods, here the pressure field is only calculated at the specific discrete times of interest; and it does not affect the computation of the flow dynamics.

3.4. References

Bell, JB, Colella, P & Glaz, HM 1989, 'A second-order projection method for the incompressible Navier-Stokes equations', *Journal of Computational Physics*, vol. 85, pp. 257–283.

Bollada, PC & Phillips, TN 2012, 'On the mathematical modelling of a compressible viscoelastic fluid', *Archive for Rational Mechanics and Analysis*, vol. 205, pp. 1–26.

Brezzi, F & Douglas Jr, J 1988, 'Stabilized mixed methods for the Stokes problem', *Numerische Mathematik*, vol. 53, pp. 225–235.

Chorin, AJ 1967, 'A numerical method for solving incompressible viscous flow problems', *Journal of Computational Physics*, vol. 2, pp. 12–26.

Chorin, AJ 1968, 'Numerical solution of the Navier-Stokes equations', *Mathematics of Computation*, vol. 22, pp. 745–762.

Dukowicz, JK & Dvinsky, AS 1992, 'Approximate factorization as a high order splitting for the implicit incompressible flow equations', *Journal of Computational Physics*, vol. 102, pp. 336–347.

Guermond, JL, Minev, P & Shen, J 2006, 'An overview of projection methods for incompressible flows', *Computer Methods in Applied Mechanics and Engineering*, vol. 195, pp. 6011–6045.

Guj, G & Stella, F 1993, 'A vorticity-velocity method for the numerical solution of 3D incompressible flows', *Journal of Computational Physics*, vol. 106, pp. 286–298.

Henriksen, MO & Holmen, J 2002, 'Algebraic splitting for incompressible Navier–Stokes equations', *Journal of Computational Physics*, vol. 175, pp. 438–453.

Hughes, TJ, Liu, WK & Brooks, A 1979, 'Finite element analysis of incompressible viscous flows by the penalty function formulation', *Journal of Computational Physics*, vol. 30, pp. 1–60.

Issa, RI 1986, 'Solution of the implicitly discretised fluid flow equations by operator-splitting', *Journal of Computational Physics*, vol. 62, pp. 40–65.

Kim, J & Moin, P 1985, 'Application of a fractional-step method to incompressible Navier-Stokes equations', *Journal of Computational Physics*, vol. 59, pp. 308–323.

Kupferman, R 2001, 'A central-difference scheme for a pure stream function formulation of incompressible viscous flow', *SIAM Journal on Scientific Computing*, vol. 23, pp. 1–18.

Kwon, Y 2012, 'Numerical description of elastic flow instability and its dependence on liquid viscoelasticity in planar contraction', *Journal of Rheology*, vol. 56, pp. 1335–1362.

Lo, DC, Murugesan, K & Young, DL 2005, 'Numerical solution of three-dimensional velocity–vorticity Navier–Stokes equations by finite difference method', *International Journal for Numerical Methods in Fluids*, vol. 47, pp. 1469–1487.

Meitz, HL & Fasel, HF 2000, 'A compact-difference scheme for the Navier–Stokes equations in vorticity–velocity formulation', *Journal of Computational Physics*, vol. 157, pp. 371–403.

Ni, MJ & Abdou, MA 2007, 'A bridge between projection methods and SIMPLE type methods for incompressible Navier–Stokes equations', *International Journal for Numerical Methods in Engineering*, vol. 72, pp. 1490–1512.

Oliveira, PJ 2000, 'A traceless stress tensor formulation for viscoelastic fluid flow', *Journal of Non-Newtonian Fluid Mechanics*, vol. 95, pp. 55–65.

Patankar, S 1980, *Numerical heat transfer and fluid flow*, CRC Press.

Perot, JB 1993, 'An analysis of the fractional step method', *Journal of Computational Physics*, vol. 108, pp. 51–58.

Qian, Z, Zhang, J & Li, C 2010, 'Preconditioned pseudo-compressibility methods for incompressible Navier-Stokes equations', *Science China Physics, Mechanics and Astronomy*, vol. 53, pp. 2090–2102.

Quarteroni, A, Saleri, F & Veneziani, A 2000, 'Factorization methods for the numerical approximation of Navier–Stokes equations', *Computer Methods in Applied Mechanics and Engineering*, vol. 188, pp. 505–526.

Shen, J 1992, 'On pressure stabilization method and projection method for unsteady Navier-Stokes equations', in: *Advances in Computer Methods for Partial Differential Equations*, eds R Vichnevetsky, D Knight & G Richter, pp. 658–662.

Shen, J 1995, 'On error estimates of the penalty method for unsteady Navier-Stokes equations', *SIAM Journal on Numerical Analysis*, vol. 32, pp. 386–403.

Shen, J 1996, 'On a new pseudocompressibility method for the incompressible Navier-Stokes equations', *Applied Numerical Mathematics*, vol. 21, pp. 71–90.

Shen, J 1997, 'Pseudo-compressibility methods for the unsteady incompressible Navier-Stokes equations', in: *Proceedings of the 1994 Beijing symposium on nonlinear evolution equations and infinite dynamical systems*, pp. 68–78.

Shukla, RK, Tatineni, M & Zhong, X 2007, 'Very high-order compact finite difference schemes on non-uniform grids for incompressible Navier–Stokes equations', *Journal of Computational Physics*, vol. 224, pp. 1064–1094.

Sousa, FS, Oishi, CM & Buscaglia, GC 2015, 'Spurious transients of projection methods in microflow simulations', *Computer Methods in Applied Mechanics and Engineering*, vol. 285, pp. 659–693.

Versteeg, HK & Malalasekera, W 2007, *An introduction to computational fluid dynamics: the finite volume method*, Pearson Education.

Weinan, E & Liu, JG 1997, 'Finite difference methods for 3D viscous incompressible flows in the vorticity–vector potential formulation on nonstaggered grids', *Journal of Computational Physics*, vol. 138, pp. 57–82.

Weinan, E & Liu, JG 1996, 'Vorticity boundary condition and related issues for finite difference schemes', *Journal of Computational Physics*, vol. 124, pp. 368–382.

Weinan, E & Liu, JG 1996, 'Essentially compact schemes for unsteady viscous incompressible flows', *Journal of Computational Physics*, vol. 126, pp. 122–138.

Xue, SC, Tanner, RI & Phan-Thien, N 2004, 'Numerical modelling of transient viscoelastic flows' *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 33–58.

Chapter 4

Discretization methods

This chapter presents the discretization methods used to solve the constitutive equations and the conservation laws presented in the previous chapters. The streamfunction evolution equation is discretized with the discrete streamfunction method of Chang, Giraldo and Perot (2002). Directly discretizing the streamfunction equation is a tedious task because of the fourth-order derivatives. Instead, we apply discrete curl operators to discretize the momentum equations, in order to obtain a discretized system of equations for the streamfunction. The momentum and constitutive equations are discretized with the finite–volume method on a staggered Cartesian grid. This chapter also presents an iterative solution algorithm and an adaptive time–stepping procedure that improve the efficiency of the algorithm. This solution algorithm that solves the discretized governing equations has been implemented in Matlab and it was used to obtain the numerical results presented in chapter 6. Finally, the potential extension of the method to three-dimensions is discussed.

4.1. Discrete curl operators

Let the linear matrix system for the discretized momentum equations be:

$$\mathbf{A}\mathbf{u} = \mathbf{b}\,, \tag{4.1}$$

where $\mathbf{u}$ is the solution vector of the discrete velocity variables, $\mathbf{A}$ is the Jacobi matrix of the discrete velocity variables in the discretized momentum equations, and $\mathbf{b}$ is the source term vector. The pressure gradients do not need to be included in the system of equations (4.1) since they vanish with the curl transformation. A linear system of equations for the vorticity equation is first obtained by applying the discrete curl operator $\mathbf{R}$, which relates the solution vector of the discrete velocities to the solution vector $\boldsymbol{\omega}$ of discrete vorticity variables, to the discretized momentum equations:

$$\mathbf{R}\mathbf{A}\mathbf{u} = \mathbf{R}\mathbf{b}\,, \tag{4.2}$$

with

$$\boldsymbol{\omega} = \mathbf{R}\mathbf{u}\,. \tag{4.3}$$

The curl operator $\mathbf{R}$ results in the integration of the momentum equations along closed lines around the discrete streamfunction variables. Secondly, the linear matrix system for the vorticity equation is expressed in terms of discrete streamfunction variables, by the following change of variable:

$$\mathbf{u} = \mathbf{C}\boldsymbol{\Phi}\,, \tag{4.4}$$

where $\mathbf{C}$ is the discretized curl operator relating to the discrete velocity variables to the vector of the discrete streamfunction variables $\mathbf{\Phi}$. Note that $\mathbf{R}$ and $\mathbf{C}$ are two different curl operators. Finally these transformations yield a linear matrix system expressing the vorticity equation in terms of discrete streamfunction variables:

$$\mathbf{RAC\Phi} = \mathbf{Rb}\,, \tag{4.5}$$

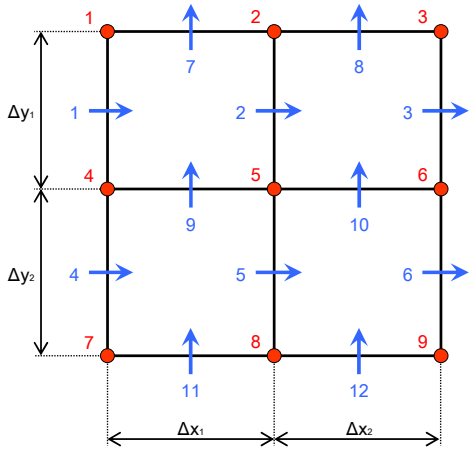
where the matrix products $\mathbf{RAC}$ and $\mathbf{Rb}$ are the Jacobi matrix and the source term vector, respectively.

The discrete curl operators $\mathbf{R}$ and $\mathbf{C}$ are discretized using finite-differences, taking advantage of a staggered variable arrangement on a Cartesian grid, see Figure 4.1. Discrete streamfunction values are located at the grid intersections. The normal velocity components lying on the cell's edges, evaluated from equation (4.4), correspond to the face-average values. Figure 4.1 shows how the discrete curl operators $\mathbf{R}$ and $\mathbf{C}$ are constructed for a Cartesian grid, but the procedure is essentially the same for an unstructured grid, see the paper of Chang, Giraldo and Perot (2002) for more details. It is important to notice that the discrete curl operators are discretized in a way such that the null vector calculus identities $\nabla\times(\nabla a)=0,\ \forall a\in\mathbb{R}$ and $\nabla\cdot(\nabla\times\mathbf{A})=0,\ \forall\mathbf{A}\in\mathbb{R}^3$ are fulfilled on the discrete level:

$$\mathbf{RG} = \mathbf{0}\,, \tag{4.6}$$

$$\mathbf{DC} = \mathbf{0}\,, \tag{4.7}$$

where $\mathbf{G}$ and $\mathbf{D}$ are the discrete gradient and divergence operators of the discretized momentum and continuity equations. These relations ensure the elimination of the pressure unknowns in the conservation equations, and the automatic enforcement of the continuity constraint on the discrete level to the machine precision[1].



$$\mathbf{C}=\begin{bmatrix}\mathbf{C}_x\\ \mathbf{C}_y\end{bmatrix}=\left[\begin{array}{ccccc}\mathbf{\Lambda y}_{cv}^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{\Lambda y}_{cv}^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda x}_{cv}^{-1} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda x}_{cv}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda x}_{cv}^{-1}\end{array}\right]\left[\begin{array}{ccc|ccc|ccc}1&0&0&-1&0&0&0&0&0\\0&1&0&0&-1&0&0&0&0\\0&0&1&0&0&-1&0&0&0\\\hline 0&0&0&1&0&0&-1&0&0\\0&0&0&0&1&0&0&-1&0\\0&0&0&0&0&1&0&0&-1\\\hline 1&-1&0&0&0&0&0&0&0\\0&1&-1&0&0&0&0&0&0\\\hline 0&0&0&1&-1&0&0&0&0\\0&0&0&0&1&-1&0&0&0\\\hline 0&0&0&0&0&0&1&-1&0\\0&0&0&0&0&0&0&1&-1\end{array}\right]$$

$$\mathbf{\Lambda x}_{cv}^{-1}=\operatorname{diag}\left(\Delta x_1^{-1},\Delta x_2^{-1}\right)$$

$$\mathbf{\Lambda y}_{cv}^{-1}=\operatorname{diag}\left(\Delta y_1^{-1},\Delta y_2^{-1}\right)$$

$$\mathbf{\Lambda x}_{cn}=\tfrac{1}{2}\operatorname{diag}\left(\Delta x_1,\Delta x_1+\Delta x_2,\Delta x_2\right)$$

$$\mathbf{\Lambda y}_{cn}=\tfrac{1}{2}\operatorname{diag}\left(\Delta y_1,\Delta y_1+\Delta y_2,\Delta y_2\right)$$

$$\mathbf{R}=\begin{bmatrix}\mathbf{R}_x & \mathbf{R}_y\end{bmatrix}=\left[\begin{array}{cccccc|cccccc}1&0&0&0&0&0&1&0&0&0&0&0\\0&1&0&0&0&0&-1&1&0&0&0&0\\0&0&1&0&0&0&0&-1&0&0&0&0\\\hline -1&0&0&1&0&0&0&0&1&0&0&0\\0&-1&0&0&1&0&0&0&-1&1&0&0\\0&0&-1&0&0&1&0&0&0&-1&0&0\\\hline 0&0&0&-1&0&0&0&0&0&0&1&0\\0&0&0&0&-1&0&0&0&0&0&-1&1\\0&0&0&0&0&-1&0&0&0&0&0&-1\end{array}\right]\left[\begin{array}{ccccc}\mathbf{\Lambda x}_{cn} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{\Lambda x}_{cn} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{\Lambda y}_{cn} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda y}_{cn} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Lambda y}_{cn}\end{array}\right]$$

Figure 4.1: Staggered variable arrangement, numbering of the unknowns, and finite-difference construction of the discrete curl operators $\mathbf{R}$ and $\mathbf{C}$, in a 2D geometry.

[1] In practice, there is no need to discretize the gradient and divergence operators to solve the streamfunction matrix system (4.5); therefore the discrete null vector calculus identities (4.6) and (4.7) are actually mental constructions.

Note that the line-integrations around the discrete streamfunction variables laying on the boundary are incomplete; hence these streamfunctions are rather determined by boundary conditions. The essential streamfunction boundary conditions are obtained from the integration of the normal velocities over the boundary. Using the definition of the streamfunction (as the vector potential of the velocity) and the Kelvin–Stokes theorem of vector calculus yields a contour integral:

$$\iint_{\partial\Omega} \boldsymbol{u}\cdot\hat{\boldsymbol{n}}\,\mathrm{d}S = \iint_{\partial\Omega} (\nabla\times\boldsymbol{\Phi})\cdot\hat{\boldsymbol{n}}\,\mathrm{d}S = \oint_{\Sigma} \boldsymbol{\Phi}\cdot d\boldsymbol{r} \tag{4.8}$$

where Σ is the contour of the boundary surface $\partial\Omega$. In the two-dimensional case, the surface integral of the normal velocity becomes a line integral along the boundary, and the contour integral of the streamfunction turns into an oriented difference, see Figure 4.2. Hence, the streamfunction boundary condition along the boundary segment between two points $\mathrm{p1}$ and $\mathrm{p2}$ is given by:

$$\begin{aligned} \phi_{\mathrm{p1}} - \phi_{\mathrm{p2}} &= \int_{\mathrm{p1}}^{\mathrm{p2}} v(x)\,\mathrm{d}x, \quad \text{along a horizontal boundary,} \\ \phi_{\mathrm{p2}} - \phi_{\mathrm{p1}} &= \int_{\mathrm{p2}}^{\mathrm{p1}} u(y)\,\mathrm{d}y, \quad \text{along a vertical boundary.} \end{aligned} \tag{4.9}$$

This yields constant values of the streamfunction along the walls and the symmetry lines. At the inlet boundaries, the line integral of the normal velocity is calculated either analytically or numerically. At the outlets, where the normal velocity is an unknown, the streamfunction boundary condition is derived from the fully-developed Neumann boundary condition $\partial u_{\mathrm{n}}/\partial x_{\mathrm{n}}\big|_{\partial\Omega} = 0$, which gives:

$$\partial\phi/\partial x_{\mathrm{n}}\big|_{\partial\Omega} = 0\,. \tag{4.10}$$

In addition, since the streamfunction is a potential field, a reference value needs to be specified at a reference point on the boundary, in order to set the streamfunction level:

$$\phi(\boldsymbol{x}_{\mathrm{ref}}) = \phi_{\mathrm{ref}}, \ \text{where} \ \boldsymbol{x}_{\mathrm{ref}} \in \partial\Omega, \tag{4.11}$$

for instance, with a null value on a symmetry line or wall boundary. Finally, the boundary conditions (4.9)–(4.11) replace the incomplete equations for the boundary streamfunction variables (see Figure 4.3) in the matrix system (4.5). Natural boundary conditions for the tangential velocity to the boundaries are imposed through the source terms in the matrix system (4.1), with the same procedure as the one described by Perot (1993).

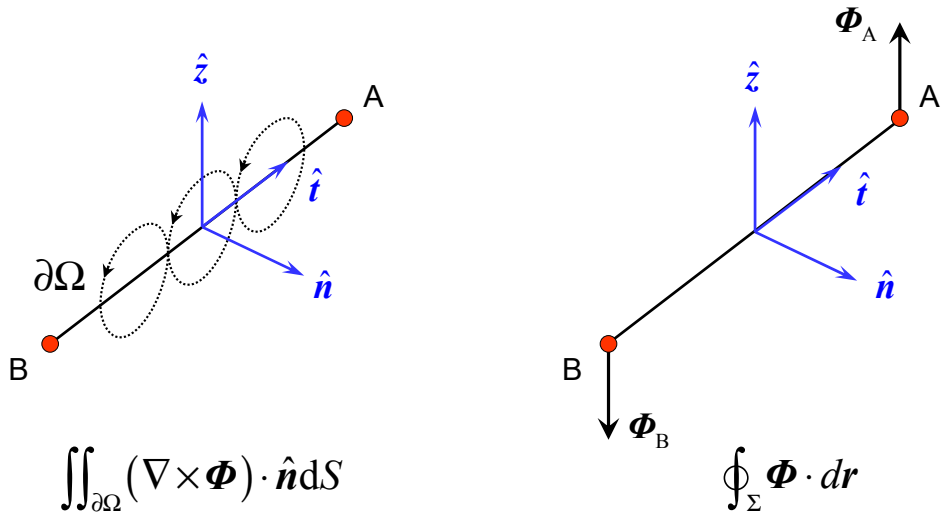


Figure 4.2: Kelvin–Stokes theorem applied on the surface integral of the boundary, in two-dimensions.

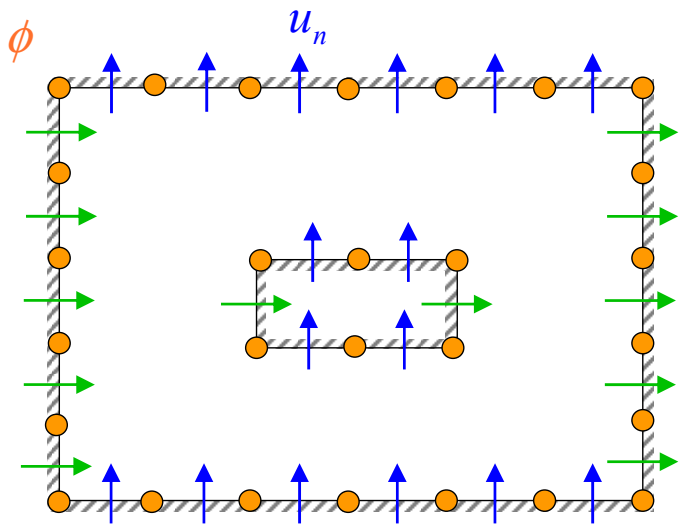


Figure 4.3: Streamfunction variables laying on the boundary, governed by the surface integrals (or line integral in 2D) of the normal velocities to the boundary, rather than the vorticity conservation discretized with the discrete curl operators.

4.2. Finite-volume discretization

In this subsection, we describe the finite-volume discretization of a generic convection–diffusion equation with the form:

$$\frac{\partial \chi}{\partial t} + u_k \frac{\partial \chi}{\partial x_k} = \Gamma_\chi \frac{\partial^2 \chi}{\partial x_k \partial x_k} + S_\chi \,, \tag{4.12}$$

which can either be a projection of the momentum equation or the log–conformation equation, depending on the interpretation of χ, Γ_χ and S_χ; see Table 4.1. Equation (4.12) is integrated over a control volume V and the divergence theorem is applied to the convective and diffusive terms:

$$\int_V \frac{\partial \chi}{\partial t} \mathrm{d}V + \oint_{\partial V} \left(\boldsymbol{u}\chi - \Gamma_\chi \frac{\partial \chi}{\partial \hat{\boldsymbol{n}}} \right) \cdot \hat{\boldsymbol{n}} \mathrm{d}S = \int_V S_\chi \mathrm{d}V \,, \tag{4.13}$$

yielding a flux balance between the faces f of the control volume:

$$\frac{\partial \chi_V}{\partial t} + \frac{1}{V} \sum_{\mathrm{f}} \left(u_{\mathrm{f}} \chi_{\mathrm{f}} - \Gamma_\chi \left. \frac{\partial \chi}{\partial x_n} \right|_{\mathrm{f}} \right) = S_{\chi,V} \,, \tag{4.14}$$

where χ_V and $S_{\chi,V}$ are the volume-average of the transported quantity and the source term, inside the control volume. The evaluation of the fluxes requires an interpolation of the quantities χ_{f} at the faces of the control volumes, as well as their gradients $\left. \partial\chi/\partial x_n \right|_{\mathrm{f}}$ in the normal directions. The latter gradients, which are related to the diffusive fluxes, are evaluated with the second-order central–differences of the volume-average quantities in the neighboring cells.

Equations	χ	Γ_χ	S_χ
i^{th}–component of the momentum	ρu_i	$\beta\eta_0/\rho$	$\dfrac{(1-\beta)\eta_0}{\lambda}\left[f_{\text{S}}(\boldsymbol{c})\right]_{ij}$
ij^{th}–component of the log–conformation	$\boldsymbol{\Psi}_{ij}$	0	$\left(\boldsymbol{\Omega\Psi}-\boldsymbol{\Psi\Omega}\right)_{ij}+2E_{ij}-\dfrac{1}{\lambda}\left\{\exp(-\boldsymbol{\Psi})\,f_{\text{R}}\left[\exp(\boldsymbol{\Psi})\right]\right\}_{ij}$

Table 4.1: Interpretation of the terms χ, Γ_χ and S_χ in the generic convection–diffusion equation (4.12), for the momentum and log–conformation equations.

It has been shown by Alves, Pinho and Oliveira (2000) as well as Min, Yoo and Choi (2001) that the spatial discretization scheme of the viscoelastic stress fluxes has a significant impact on the numerical solution. In this work, the face-averaged values χ_{f} are obtained with the CUBISTA interpolation scheme, introduced by Alves, Oliveira and Pinho (2003), which consists of a quadratic upwind scheme, with total-variation-diminishing flux limiters promoting the numerical stability. The CUBISTA scheme was specially developed to enhance the iterative convergence of viscoelastic flow solvers. For convenience, the interpolation scheme is expressed in terms of the normalized variable and space formulation, where

$$\tilde{\chi}_{\text{f}}=\frac{\chi_{\text{f}}-\chi_{\text{UU}}}{\chi_{\text{D}}-\chi_{\text{UU}}},\ \tilde{\chi}_{\text{U}}=\frac{\chi_{\text{U}}-\chi_{\text{UU}}}{\chi_{\text{D}}-\chi_{\text{UU}}} \tag{4.15}$$

are the normalized averaged-values of the transported quantities at the CV's face, and inside the upwind cell, respectively, and

$$\tilde{\xi}_{\text{f}}=\frac{\xi_{\text{f}}-\xi_{\text{UU}}}{\xi_{\text{D}}-\xi_{\text{UU}}},\ \tilde{\xi}_{\text{U}}=\frac{\xi_{\text{U}}-\xi_{\text{UU}}}{\xi_{\text{D}}-\xi_{\text{UU}}} \tag{4.16}$$

are the normalized space variables, with subscripts "D", "U" and "UU" referring to volume-average values in the downstream, first upstream and second upstream control volumes, respectively, see Figure 4.4. The interpolation $\tilde{\chi}_{\text{f}}\left(\tilde{\chi}_{\text{U}},\tilde{\xi}_{\text{U}},\tilde{\xi}_{\text{f}}\right)$ with the CUBISTA scheme reads:

$$\tilde{\chi}_{\text{f}}\left(\tilde{\chi}_{\text{U}},\tilde{\xi}_{\text{U}},\tilde{\xi}_{\text{f}}\right)=\begin{cases}\left[1+\dfrac{\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}}{3\left(1-\tilde{\xi}_{\text{U}}\right)}\right]\dfrac{\tilde{\xi}_{\text{f}}}{\tilde{\xi}_{\text{U}}}\tilde{\chi}_{\text{U}}, & \text{if}\quad 0<\tilde{\chi}_{\text{U}}<\dfrac{3}{4}\tilde{\xi}_{\text{U}}\\ \dfrac{\tilde{\xi}_{\text{f}}\left(1-\tilde{\xi}_{\text{f}}\right)}{\tilde{\xi}_{\text{U}}\left(1-\tilde{\xi}_{\text{U}}\right)}\tilde{\chi}_{\text{U}}+\dfrac{\tilde{\xi}_{\text{f}}\left(\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}\right)}{1-\tilde{\xi}_{\text{U}}}, & \text{if}\quad \dfrac{3}{4}\tilde{\xi}_{\text{U}}\le\tilde{\chi}_{\text{U}}\le\dfrac{1+2\left(\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}\right)}{2\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}}\tilde{\xi}_{\text{U}}\\ 1-\dfrac{1-\tilde{\xi}_{\text{f}}}{2\left(1-\tilde{\xi}_{\text{U}}\right)}\left(1-\tilde{\chi}_{\text{U}}\right), & \text{if}\quad \dfrac{1+2\left(\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}\right)}{2\tilde{\xi}_{\text{f}}-\tilde{\xi}_{\text{U}}}\tilde{\xi}_{\text{U}}<\tilde{\chi}_{\text{U}}<1\\ \tilde{\chi}_{\text{U}}, & \text{otherwise}\end{cases} \tag{4.17}$$

The time derivative $\partial\chi_V/\partial t$ is discretized implicitly with the two-level backward differentiation formula (BDF2):

$$\left.\frac{\partial\chi_V}{\partial t}\right|^n=\frac{1}{\Delta t^n}\left(\frac{2\varpi+1}{\varpi+1}\chi_V^n-(\varpi+1)\chi_V^{n-1}+\frac{\varpi^2}{\varpi+1}\chi_V^{n-2}\right)+O\left(\Delta t^2\right), \tag{4.18}$$

where the subscript n refers the value at the current time t^n, the superscripts n–1 and n–2 refer to the values at the two previous discrete times t^{n-1} and t^{n-2}, $\Delta t^n=t^n-t^{n-1}$ is the time-step increment, and $\varpi=\Delta t^n/\Delta t^{n-1}$ is the

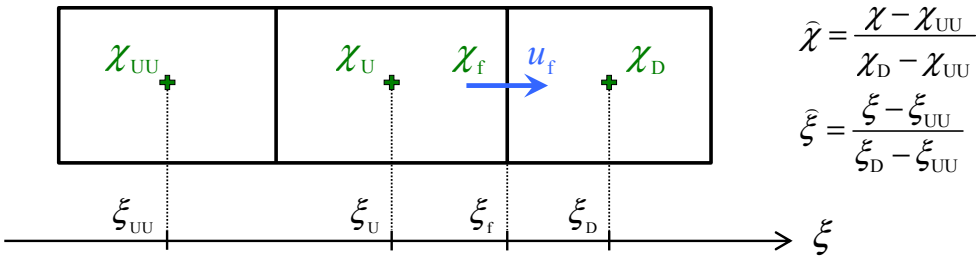


Figure 4.4: Definition of the local variables and local coordinates in the upwind and downwind cells, related to interpolation $\hat{\chi}_f\left(\hat{\chi}_U, \hat{\xi}_U, \hat{\xi}_f\right)$ at a faces of the control volumes.

time-step increment ratio. The BDF2 scheme is *A-stable* and second-order accurate, which makes it suitable to solve stiff hyperbolic equations, with a good compromise between numerical stability and accuracy.

The quadratic non-linear terms for the inertia, in the momentum equation, are linearized using a second-order Taylor expansion:

$$\begin{aligned}\left(u_i u_j\right)^n &= \left(u_i u_j\right)^{n-1} + \left.\frac{\partial\left(u_i u_j\right)}{\partial u_k}\right|^{n-1}\left(u_k^n - u_k^{n-1}\right) + O\left(\Delta t^2\right) \\ &= u_i^n u_j^{n-1} + u_i^{n-1} u_j^n - u_i^{n-1} u_j^{n-1} + O\left(\Delta t^2\right).\end{aligned} \tag{4.19}$$

This approximation is suitable for approximating inertia terms for flows at low and moderate Reynolds numbers. As an example, Kim, Baek and Sung (2002) were able to solve accurately turbulent Newtonian flows at $Re = 4200$, with the same implicit linearization.

The divergence of the viscoelastic stress-tensor, in the source term of the momentum equations, is also approximated with finite-differences. Note that all the normal and shear components of the extra-stress are located at the center of the control volumes, similarly as the components of the log–conformation tensor. The classical finite-difference schemes take advantage of the staggered variable arrangement, by assuming linear variations of the viscoelastic extra-stresses inside the control volumes, as in the work of Fattal and Kupferman (2005):

$$\frac{\partial \sigma_{xx}}{\partial x} = \frac{\sigma_{xx,E} - \sigma_{xx,W}}{\left(\Delta x_E + \Delta x_W\right)/2}, \tag{4.20}$$

$$\frac{\partial \sigma_{xy}}{\partial y} = \frac{1}{2}\frac{\sigma_{xy,NW} + \sigma_{xy,NE} - \sigma_{xy,SW} - \sigma_{xy,SE}}{\Delta y_N + 2\Delta y + \Delta y_S}, \tag{4.21}$$

where the different subscripts refer to the different neighboring extra-stresses of the velocity's control volume, see Figure 4.5. This classical scheme is hereinafter referred to as the LIN scheme. Here, we suggest a higher-order scheme assuming linear variations of the gradient of the viscoelastic extra-stress; hence quadratic variations of the viscoelastic extra-stress. This scheme is referred to as the QUAD scheme and the volume-averages of the stress-divergences are obtained with the following arithmetic averages:

$$\frac{\partial \sigma_{xx}}{\partial x} = \frac{\Delta x_E \left.\frac{\partial \sigma_{xx}}{\partial x}\right|_W + \Delta x_W \left.\frac{\partial \sigma_{xx}}{\partial x}\right|_E}{\Delta x_E + \Delta x_W}, \tag{4.22}$$

$$\frac{\partial \sigma_{xy}}{\partial y} = \frac{(\Delta y + \Delta y_S)\left(\Delta x_E \left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{NW} + \Delta x_W \left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{NE}\right) + (\Delta y_N + \Delta y)\left(\Delta x_E \left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{SW} + \Delta x_W \left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{SE}\right)}{(\Delta y_N + 2\Delta y + \Delta y_S)(\Delta x_E + \Delta x_W)}, \quad (4.23)$$

where

$$\left.\frac{\partial \sigma_{xx}}{\partial x}\right|_W = \frac{1}{\Delta x_{WW} + 2\Delta x_W + \Delta x_E}\left[(\Delta x_W + \Delta x_{WW})\frac{\sigma_{xx,E} - \sigma_{xx,W}}{(\Delta x_E + \Delta x_W)/2} + (\Delta x_E + \Delta x_W)\frac{\sigma_{xx,W} - \sigma_{xx,WW}}{(\Delta x_W + \Delta x_{WW})/2}\right], \quad (4.24)$$

$$\left.\frac{\partial \sigma_{xx}}{\partial x}\right|_E = \frac{1}{\Delta x_W + 2\Delta x_E + \Delta x_{EE}}\left[(\Delta x_E + \Delta x_W)\frac{\sigma_{xx,EE} - \sigma_{xx,E}}{(\Delta x_{EE} + \Delta x_E)/2} + (\Delta x_{EE} + \Delta x_E)\frac{\sigma_{xx,E} - \sigma_{xx,W}}{(\Delta x_E + \Delta x_W)/2}\right], \quad (4.25)$$

$$\left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{NW} = \frac{\sigma_{xy,NW} - \sigma_{xy,W}}{(\Delta y_N + \Delta y)/2}, \quad (4.26)$$

$$\left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{NE} = \frac{\sigma_{xy,NE} - \sigma_{xy,E}}{(\Delta y_N + \Delta y)/2}, \quad (4.27)$$

$$\left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{SW} = \frac{\sigma_{xy,W} - \sigma_{xy,SW}}{(\Delta y + \Delta y_S)/2}, \quad (4.28)$$

$$\left.\frac{\partial \sigma_{xy}}{\partial y}\right|_{SE} = \frac{\sigma_{xy,E} - \sigma_{xy,WE}}{(\Delta y + \Delta y_S)/2}, \quad (4.29)$$

are the finite-difference approximations of the viscoelastic extra-stress gradient, using the same staggered variable arrangement as in Figure 4.5.

The nonlinearities in the constitutive equations, due to the material-derivative and the matrix-exponential of the log–conformation tensor, are treated explicitly. As a result, each evolution equation for a variable χ_V^n is discretized into the algebraic form

$$a_0^\chi \chi_V^n = \sum_{nb} a_{nb}^\chi \chi_{nb}^n + b^\chi, \quad (4.30)$$

where χ_{nb}^n are unknown values in the neighboring control volumes, a_0^χ, a_{nb}^χ and b^χ are coefficients containing explicit expressions of the solutions of the previous time-step. The contributions of the source terms are included in b^χ.

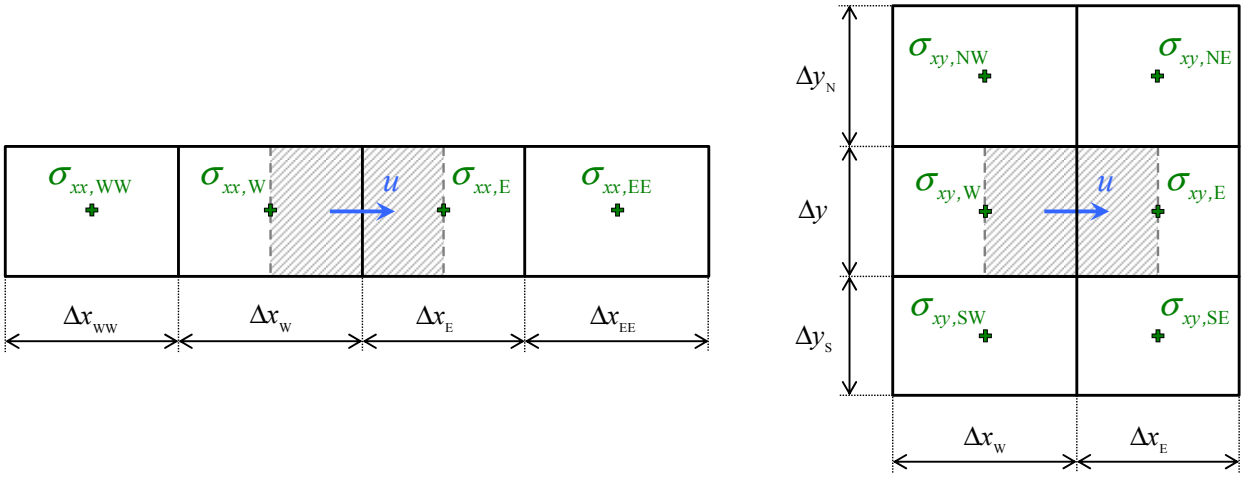


Figure 4.5: Staggered variable arrangement used for the finite-difference approximation of the divergence of the viscoelastic extra-stress inside the hatched control volume.

The explicit representation of the covariant term $\boldsymbol{\Omega\Psi} - \boldsymbol{\Psi\Omega}$ decouples the three components of the log–conformation tensor. Therefore, the two-dimensional streamfunction/log–conformation scheme yields four matrix systems: one for the streamfunction field, and one for each of the three components of the log–conformation tensor that are solved in parallel. Finally, the streamfunction and the log–conformation tensor are solved sequentially, with successive substitution iterations until the convergence of the nonlinearities.

4.3. Solution algorithm

This subsection describes the procedure that solves the coupled discretized system of equations for the constitutive models (reformulated with the log–conformation representation) and the conservation laws (expressed in terms of the streamfunction). We assume that discrete temporal solutions are known until the $(n-1)^{\text{th}}$ time-step. The following solution algorithm calculates the next discrete temporal solution, at the n^{th} time-step. Provisional guessed solution vectors $\boldsymbol{\Phi}_0^n$ and $\boldsymbol{\Psi}_0^n$, respectively for the streamfunctions and the components of the log–conformation tensor, are evaluated by linear extrapolation from the results of the previous time-steps. The provisional guessed solution for the velocity vector is $\mathbf{u}_0^n = \mathbf{C}\boldsymbol{\Phi}_0^n$. Then, the solution algorithm computes iteratively the components of the new log–conformation tensor and the new streamfunctions.

Since the governing equations are discretized with the implicit BDF2 scheme, there is no formal restriction on the time-step size. However, the successive substitution iterations can have convergence issues when the Courant number exceeds unity. The convergence of the solution algorithm is enhanced with the under-relaxation of the solution iterates. The log–conformation tensor contributes linearly to the streamfunction equations, thus the convergence of the log–conformation tensor results in the convergence of the streamfunction. In contrast, the velocity has a non-linear effect on the evolution of the log–conformation tensor. Therefore, the under-relaxation is only applied to the log–conformation iterates. We use an adaptive under-relaxation, the Aitken's delta-squared process, which improves the convergence rate of a series of solution vectors. At each k^{th} iteration, the Aitken's relaxation factor α_k is evaluated as:

$$\alpha_k = \alpha_{k-1}\left(1 + \frac{\left(\Delta\tilde{\boldsymbol{\Psi}}_{k-1} - \Delta\tilde{\boldsymbol{\Psi}}_k\right)^{\mathrm{T}} \cdot \Delta\tilde{\boldsymbol{\Psi}}_k}{\left(\Delta\tilde{\boldsymbol{\Psi}}_{k-1} - \Delta\tilde{\boldsymbol{\Psi}}_k\right)^2}\right), \tag{4.31}$$

with

$$\Delta\tilde{\boldsymbol{\Psi}}_k = \boldsymbol{\Psi}_{k-1}^n - \tilde{\boldsymbol{\Psi}}_k^n, \tag{4.32}$$

where $\tilde{\boldsymbol{\Psi}}_k^n$ is the latest non-under-relaxed iterate vector of the log–conformation solution, and $\boldsymbol{\Psi}_{k-1}^n$ is the Log–conformation vector of the previous under-relaxed solution iterate. In general, adaptive under-relaxation is more efficient than under-relaxation with a fixed relaxation factor. Since the Aitken's adaptive under-relaxation requires two previous solution iterates, the under-relaxation is activated from the third iteration.

In summary, the solution algorithm at the k^{th} iteration proceeds as follows:

1. Compute the new solution vector $\tilde{\boldsymbol{\Psi}}_k^n$ of the discretized system of equations for each component of the log–conformation tensor:

$$\begin{aligned}\frac{1}{\Delta t^n}\frac{2\varpi+1}{\varpi+1}\tilde{\boldsymbol{\Psi}}_k^n + \mathbf{u}_{k-1}^n \cdot \nabla\tilde{\boldsymbol{\Psi}}_k^n = \frac{1}{\Delta t^n}\left((\varpi+1)\boldsymbol{\Psi}^{n-1} - \frac{\varpi^2}{\varpi+1}\boldsymbol{\Psi}^{n-2}\right)\\ + \left(\boldsymbol{\Omega}\left(\mathbf{u}_{k-1}^n, \boldsymbol{\Psi}_{k-1}^n\right)\boldsymbol{\Psi}_{k-1}^n - \boldsymbol{\Psi}_{k-1}^n\boldsymbol{\Omega}\left(\mathbf{u}_{k-1}^n, \boldsymbol{\Psi}_{k-1}^n\right)\right) + 2\mathbf{E}\left(\mathbf{u}_{k-1}^n, \boldsymbol{\Psi}_{k-1}^n\right)\\ - \frac{1}{\lambda}\exp\left(-\boldsymbol{\Psi}_{k-1}^n\right) f_{\mathrm{R}}\left[\exp\left(\boldsymbol{\Psi}_{k-1}^n\right)\right],\end{aligned} \tag{4.33}$$

where $\mathbf{u}_{k-1}^{n}$ is the velocity vector of the previous solution iterate.

2. Under-relax the solution vector of the log–conformation tensor with the relaxation factor α_k evaluated from (4.31):

$$\boldsymbol{\Psi}_k^n = \alpha_k \tilde{\boldsymbol{\Psi}}_k^n + (1-\alpha_k)\boldsymbol{\Psi}_{k-1}^n . \tag{4.34}$$

3. Recover the conformation vector $\mathbf{c}_k^n$ via the matrix-exponent of $\boldsymbol{\Psi}_k^n$:

$$\mathbf{c}_k^n = \exp\left(\boldsymbol{\Psi}_k^n\right). \tag{4.35}$$

4. Build-up the linear matrix system of the momentum equations discretized as:

$$\begin{aligned} &\frac{\rho}{\Delta t^n}\frac{2\varpi+1}{\varpi+1}\mathbf{u}_k^n + \rho\mathbf{N}\left(\mathbf{u}_k^n, \mathbf{u}^{n-1}\right) - \eta_S \mathbf{L}\left(\mathbf{u}_k^n\right) \\ &= \frac{\rho}{\Delta t^n}\left((\varpi+1)\mathbf{u}^{n-1} - \frac{\varpi^2}{\varpi+1}\mathbf{u}^{n-2} \right) + \mathbf{D}\left(G_0 f_S\left(\mathbf{c}_k^n\right)\right). \end{aligned} \tag{4.36}$$

5. Apply the discrete curl transformation to the momentum matrix system and compute the new iterate of the streamfunction vector $\boldsymbol{\Phi}_k^n$ from the linear matrix system (4.5).

6. Recover the discrete velocity vector as:

$$\mathbf{u}_k^n = \mathbf{C}\boldsymbol{\Phi}_k^n . \tag{4.37}$$

7. Evaluate the scaled norm of the residual vector of the log–conformation systems of equations, using the new solution vectors $\boldsymbol{\Psi}_k^n$ and $\mathbf{u}_k^n$:

$$r_k = \frac{\left\| \mathbf{LHS}_{\boldsymbol{\Psi}}\left(\mathbf{u}_k^n, \boldsymbol{\Psi}_k^n\right)\cdot\boldsymbol{\Psi}_k^n - \mathbf{RHS}_{\boldsymbol{\Psi}}\left(\mathbf{u}_k^n, \boldsymbol{\Psi}_k^n\right)\right\|_2}{\left\| \mathbf{LHS}_{\boldsymbol{\Psi}}\left(\mathbf{u}_k^n, \boldsymbol{\Psi}_k^n\right)\right\|_2 \cdot \left\|\boldsymbol{\Psi}_k^n\right\|_2 + \left\| \mathbf{RHS}_{\boldsymbol{\Psi}}\left(\mathbf{u}_k^n, \boldsymbol{\Psi}_k^n\right)\right\|_2}, \tag{4.38}$$

where $\mathbf{LHS}_{\boldsymbol{\Psi}}$ and $\mathbf{RHS}_{\boldsymbol{\Psi}}$ respectively are the new left-hand-side matrix and the right-hand-side vector for the log–conformation matrix system (4.33) that will be solved at the next iteration. The 2-norm of the matrix $\mathbf{LHS}_{\boldsymbol{\Psi}}$ is approximated as its largest component in absolute value $\left\|\mathbf{LHS}_{\boldsymbol{\Psi}}\right\|_2 \cong \max_{i,j}\left|\left(\mathbf{LHS}_{\boldsymbol{\Psi}}\right)_{ij}\right|$.

The iteration procedure is converged when the following termination criterion is fulfilled:

$$r_k \le \varepsilon_{\text{tol}}, \tag{4.39}$$

where ε_{tol} is a user-specified residual tolerance. The bulk of the computation resides in solving the linear matrix systems of the steps 1 and 5. The other steps of the solution algorithm consist in simple evaluations of explicit expressions.

4.4. Adaptive time-stepping

An adaptive time-stepping technique has been used in our simulation, in order to enhance computational efficiency. On the one hand, unnecessary small time-increments result in an extension of the calculation time. On the other hand, excessively large time-increments degrade the convergence of the iterations, and can cause convergence difficulties. Numerical experiments have shown that large time-increments (requiring more iterations but fewer time-steps) are preferable to short time increments, provided that convergence is reachable. Convergence difficulties occur when the series of the residuals does not decrease anymore, but oscillates around

a critical value above the prescribe residual tolerance ε_{tol}. An empirical adaptive time-stepping procedure based on the convergence rate of the residuals has been developed, according to the following principles:

(i) The time increment must increase whenever possible to speed up the computation.

(ii) The time increment must preventively decrease whenever necessary in order to avoid convergence difficulties.

(iii) In case of convergence difficulties, the calculation for the current time-step must restart with a smaller time increment.

The convergence rate of the residual between two successive iterations is defined as the ratio $R_k = r_{k+1}/r_k$. The geometric mean of the convergence rates for all the iterations within a time-step is:

$$\overline{R} = \left(\frac{r_N}{r_1} \right)^{1/(N-1)}, \tag{4.40}$$

where *N* is the total number of iterations to reach convergence. An objective convergence rate R_{obj} is specified, together with the tolerance δR_{obj}. When the value of $\overline{R}$ falls outside the interval $\left[R_{\text{obj}} - \delta R_{\text{obj}}, R_{\text{obj}} + \delta R_{\text{obj}} \right]$, the time increment for the next time-step is adjusted according to the formula:

$$\Delta t^{n+1} = \left(\frac{R_{\text{obj}}}{\overline{R}} \right)^q \Delta t^n . \tag{4.41}$$

The tolerance δR_{obj} avoids small oscillations of the time-step size at every iteration. The reactivity of the adaptive time-stepping is tuned with the exponent q. For instance, exponents larger than unity produce faster increases than decreases of the time increment. In addition, the variations of the time-step size are limited by specified maximum and minimum variation factors $f_{\min}$ and $f_{\max}$. In summary the adaptive time-stepping proceeds as follows:

- if $R_{\text{obj}} + \delta R_{\text{obj}} \le \overline{R} \le R_{\text{obj}} + \delta R_{\text{obj}}$, the time-step size is kept constant for the next time-step,
- if $\overline{R} \le R_{\text{obj}} f_{\max}^{-1/q}$, the time-step size is increased as $\Delta t^{n+1} = f_{\max} \Delta t^n$,
- if $R_{\text{obj}} f_{\min}^{-1/q} \le \overline{R}$, the time-step size is decreased as $\Delta t^{n+1} = f_{\min} \Delta t^n$,
- Otherwise, the time-step size is adapted using the formula (4.41).

Furthermore, a convergence difficulty is detected when the number of iterations within the current time-step either exceeds a specified maximum number $N_{\max}$, or $f_{\max}$ times the number of iterations of the previous time-step. In that case, the series of iterations is stopped and the calculations of the current time-step are restarted from the first iteration, with a smaller time-step size reduced by the factor $f_{\min}$. If on the contrary the iteration series converges with less iterations than a specified minimum number $N_{\min}$, then the time-step size for the next time-step is automatically increased by the factor $f_{\max}$. Finally, the time-step size itself is bounded by a maximum and a minimum value $\Delta t_{\max}$ and $\Delta t_{\min}$. Table 4.2 provides recommended parameters of the adaptive time-stepping, which are used in our simulation.

Symbols	Nomenclature	Recommended values
ε_{tol}	Residual tolerance	10^{-6}
R_{obj}	Objective convergence rate	0.5
δR_{obj}	Tolerance of the convergence rate	0.05
q	Exponent	2
$f_{\max}$	Maximum increasing factor	2
$f_{\min}$	Minimum decreasing factor	0.5
$N_{\max}$	Maximum number of iterations	40
$N_{\min}$	Minimum number of iterations	5
$\Delta t_{\max}$	Maximum time-step size	0.1
$\Delta t_{\min}$	Minimum time-step size	10^{-4}

Table 4.2: Recommended parameters for the adaptive time-stepping procedure.

4.5. Potential extension to three-dimensions

In the two-dimensional case, there is approximately the same number of streamfunction unknowns as the pressure unknown. However, the $\mathbf{RAC}$ matrix product has a better conditioning number than the Laplace operator $\mathbf{L}$ of the pressure (Chang, Giraldo & Perot 2002). Thus, if an iterative solver (for instance GMRES or BiCGSTAB) is used, the pure streamfunction system of equations will require less iteration than the pressure equation. The numerical experiments of Chang, Giraldo and Perot (2002) have shown that the solving of the pure streamfunction system of equations requires 4 to 5 time less iterations, with a Jacobi-preconditioned Conjugated–Gradient solver, than the system of pressure equations from the classical fractional-step method.

In three-dimensional case, the streamfunction vector potential has three non-zero components:

$$\boldsymbol{\Phi} = \left(\Phi_{xy},\ \Phi_{yz},\ \Phi_{zx} \right). \tag{4.42}$$

The components of the velocity vector is now defined as:

$$\begin{aligned} u &= \frac{\partial \Phi_{xy}}{\partial y} - \frac{\partial \Phi_{zx}}{\partial z}, \\ v &= \frac{\partial \Phi_{yz}}{\partial z} - \frac{\partial \Phi_{xy}}{\partial x}, \\ w &= \frac{\partial \Phi_{zx}}{\partial x} - \frac{\partial \Phi_{yz}}{\partial y}. \end{aligned} \tag{4.43}$$

In a Cartesian grid, the system of pressure equations has now three times fewer unknowns than the streamfunction system.

The main challenge of the three-dimensional calculations lies in the solving of very large matrix systems. In the current implementation, the matrix systems are solved with the built-in direct solver of Matlab, which operates the UMFPACK (Unsymmetric MultiFrontal) algorithm for sparse matrices. However, it will be preferable to solve the large matrix systems of three-dimensional problems with the iterative solvers in the Krylov subspace family, as for instance the *Conjugated–Gradient* method, BiCGSTAB, GMRES, etc. It is an open question whether or not the previous results of Chang, Giraldo and Perot (2002), about the efficiency of iterative

solvers in the two-dimensional case, still hold for the three-dimensional case. If so, then the pure streamfunction method could still be more efficient than the pressure-based methods, since the computational cost of Krylov-subspace iterative solver scales linearly with the number of unknown. (There are three times more unknowns, but the iterations converge 4 to 5 times faster). Chang, Giraldo and Perot (2002) also have noted that the streamfunction method is more efficient on unstructured tetrahedral meshes, where the full streamfunction vector is defined at each vertex (see Figure 4.6), than structured hexahedral meshes, where the streamfunction components are staggered. In both cases, each control volume has 12 streamfunction unknowns, but in the tetrahedral case, each unknown is shared by around 12 control volume, while in the structured hexahedral case, each streamfunction unknown is only shared by 4 control volumes. Therefore the number of streamfunction unknowns per cell is lower for unstructured tetrahedral meshes. Table 4.3 summarized the number of streamfunction unknowns per control volume (an unknown shared by N cells counts as $1/N$ in each cell) for the different types of meshes. For comparison purpose, pressure-based methods have 1 pressure unknown per cell, irrespective of the mesh type in 2D and 3D. Nevertheless, structured meshes results in band matrix system, which are well conditioned and generally easier to solve.

In conclusion, we cannot formally conclude whether the streamfunction method is more efficient than pressure-based fraction step methods in the three-dimensional case, but there are good chances that it is. Moreover, in the case that the discrete streamfunction method turns out not to be more efficient than the pressure-based methods, the gain in accuracy and stability due to absence of decoupling errors grants an advantage to the streamfunction formulation.

To the knowledge of the author, only one implementation of the three-dimensional pure streamfunction method has been reported in the literature (Wang & Zhang 2011; Wang, He & Zhang 2013). The three-dimensional implementation has only been done with structured Cartesian grids and the immersed boundary method, to model complex geometries.

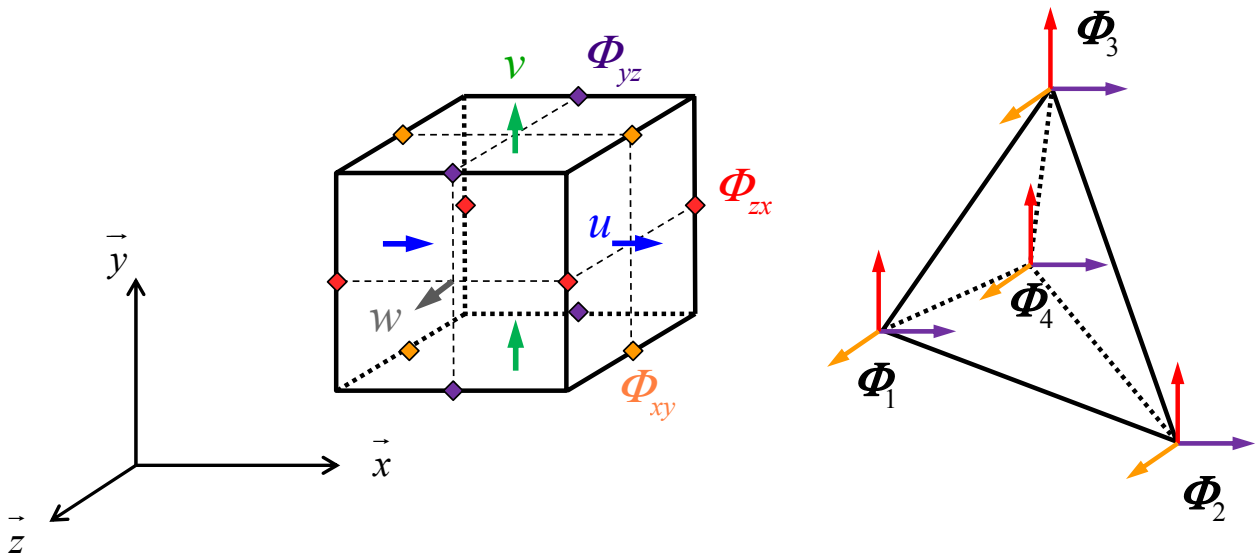


Figure 4.6: Arrangements of the streamfunction unknowns on three-dimensional Cartesian and tetrahedral meshes.

	Number of streamfunction unknowns per cell	
	Structured hexahedral mesh	Unstructured tetrahedral mesh
2D case	1	0.5
3D case	3	1

Table 4.3: Number of streamfunction unknown per cell, for different types of meshes.

4.6. References

Alves, MA, Pinho, FT & Oliveira, PJ 2000, 'Effect of a high-resolution differencing scheme on finite-volume predictions of viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 93, pp. 287–314.

Alves, MA, Oliveira, PJ & Pinho, FT 2003, 'A convergent and universally bounded interpolation scheme for the treatment of advection', *International Journal for Numerical Methods in Fluids*, vol. 41, pp. 47–75.

Chang, W, Giraldo, F & Perot, B 2002, 'Analysis of an exact fractional step method', *Journal of Computational Physics*, vol. 180, pp. 183–199.

Fattal, R & Kupferman, R 2005, 'Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 126, pp. 23–37.

Kim, K, Baek, SJ & Sung, HJ 2002, 'An implicit velocity decoupling procedure for the incompressible Navier–Stokes equations', *International Journal for Numerical Methods in Fluids*, vol. 38, pp. 125–138.

Min, T, Yoo, JY & Choi, H 2001, 'Effect of spatial discretization schemes on numerical solutions of viscoelastic fluid flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 100, pp. 27–47.

Perot, JB 1993, 'An analysis of the fractional step method', *Journal of Computational Physics*, vol. 108, pp. 51–58.

Wang, S & Zhang, X 2011, 'An immersed boundary method based on discrete stream function formulation for two-and three-dimensional incompressible flows', Journal of Computational Physics, vol. 230, pp. 3479–3499.

Wang, S, He, G & Zhang, X 2013, 'Parallel computing strategy for a flow solver based on immersed boundary method and discrete stream-function formulation', Computers & Fluids, vol. 88, pp. 210–224.

Chapter 5

Multiphase modeling

In this chapter, we first shortly present the different Eulerian surface-tracking techniques that can be used to model immiscible two-phase flows, with a special focus on the volume-of-fluid (VOF) method, which we have adopted. Afterwards, a short review on the different reconstruction algorithms is provided as well as a description of the classical directional split advection scheme and a new cellwise conservative unsplit (CCU) advection scheme. It is also showed that the proposed CCU scheme performs better than both the directional-split method and other multi-dimensional advection schemes, on classical benchmark tests. Finally, this chapter presents a new rule of mixture to average the apparent material properties of the control volume containing a mixture of the two phases. Both the new CCU scheme and rule of mixture are utilized in the simulations of viscoelastic free-surface flows in the next chapter.

5.1. Eulerian surface-tracking techniques

Common Eulerian techniques for multiphase flow simulations include the *front-tracking* methods (Tryggvason et al. 2001), the VOF method, and the *level-set* method (Osher & Fedkiw 2001; Sethian & Smereka 2003). While the front tracking methods represent explicitly the interface with geometrical objects (for instance markers, segment lines and planes), the VOF and level-set methods treat implicitly the interface, through an additional field variable, the color function. The main advantage of the implicit interface representations is that they can handle arbitrary changes of topology (Scardovelli & Zaleski 1999), while explicit interface representations require complex operations for the merge and breakup of interfaces (Zhang et al. 2001).

In the level-set method, the color function is the signed distance to the interface, governed by an advection-like equation. The advection equation is solved in a Eulerian framework using standard methods for partial differential equations, such as finite-differences, finite-volumes or finite-elements. The drawback of the level-set method is that it does not conserve the volume of each phase exactly.

We chose to use to simulate the immiscible bi-phasic flow with the VOF method, which was introduced by Hirt and Nichols (1981). In the VOF method, the presence of a certain phase at the position $\boldsymbol{x}$ and the time t is indicated by the color function $\chi(\boldsymbol{x},t)$, defined as:

$$\chi(\boldsymbol{x},t)=\begin{cases}1 & \text{if fluid no.1 is present at } (\boldsymbol{x},t),\\ 0 & \text{if fluid no.1 is not present at } (\boldsymbol{x},t).\end{cases} \tag{5.1}$$

Sharp interfaces are defined at the jump discontinuities of the color function. The evolution of the color function is governed by a transport equation:

$$\frac{\partial \chi}{\partial t} + u_i \frac{\partial \chi}{\partial x_i} = 0 \,. \tag{5.2}$$

On the discrete level, the average of the continuous color function over a control volume C gives the *liquid volume fraction* $f_C(t)$, i.e. the relative amount of fluid phase no. 1 inside C:

$$f_C(t) = \frac{1}{V_C} \int_C \chi(\boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{x} \,, \tag{5.3}$$

where V_C is the total volume of C. The cells where $0 < f < 1$ contain an interface and they are hereafter referred to as the *interfacial cells*.

Algebraic numerical methods for the resolution of hyperbolic conservation equations are not suited to solve equation (5.2), because of their inherent numerical diffusion that smoothens the jump discontinuities. The original VOF method rather uses a geometrical scheme to determine the fluxes of the liquid volume fraction through the edge of the control volume. The geometrical advection schemes require (i) an explicit geometrical reconstruction of the interface position in each interfacial cell, and (ii) an explicit evaluation of the donating region (DR) relative to each flux. The DR corresponds to the compact set of points that contains Lagrangian particles contributing to the flux, within a certain time interval (Zhang 2013). The intersection of the donating region with the reconstructed interface determines the amount of liquid volume that is advected through the edge of the control volume.

5.2. Geometrical interface reconstruction

The standard technique to represent the interface is the piecewise linear interface construction (PLIC), in which the interface inside each interfacial cell is approximated by a segment, parameterized as

$$\hat{\boldsymbol{n}} \cdot \boldsymbol{x} + d = 0 \,, \tag{5.4}$$

where $\hat{\boldsymbol{n}}$ is the normal vector of the interface pointing outward to the fluid phase no. 1, and where d is the signed distance of the interface to an origin. The geometrical interface reconstruction algorithms convert the liquid volume fractions into a set of lines representing the PLIC approximation of the interface inside all the interfacial cells, as in Figure 5.1 for instance.

There are different methods to estimate the interface normal from the distribution of the liquid volume fraction, see Figure 5.2. The simplest approximation of the interface normal is obtained by the normed gradient of the volume fraction (Youngs 1982):

$$\hat{\boldsymbol{n}} = -\frac{\nabla f}{|\nabla f|} \,, \tag{5.5}$$

evaluated by finite-differences using the values in the four neighboring cells; see Figure 5.2(a). This gradient method is first-order accurate and does not reconstruct all line interfaces exactly.

Another method, the least-square VOF interface reconstruction algorithm (LVIRA), proposed by Puckett et al. (1997), is based on the minimization of the difference between the volume below the prolonged interface into the nearest neighboring cells, and the actual volume fraction in those cells, see Figure 5.2(b). The minimization of the hatched area is a non-linear problem; therefore the LVIRA method requires an iterative optimization algorithm. The LVIRA algorithm is second-order accurate and reconstructs all line interfaces exactly.

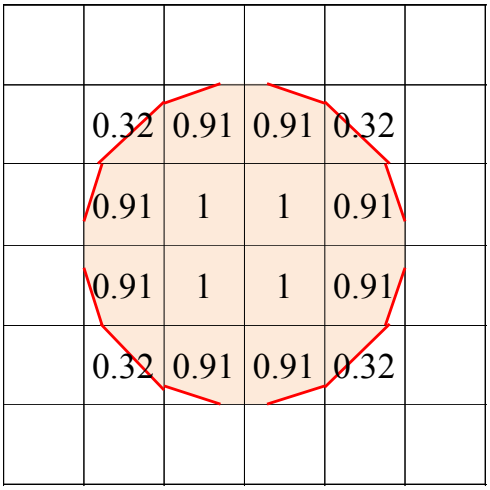


Figure 5.1: Piecewise linear interface calculation of a disk, from the liquid volume fractions on a coarse grid.

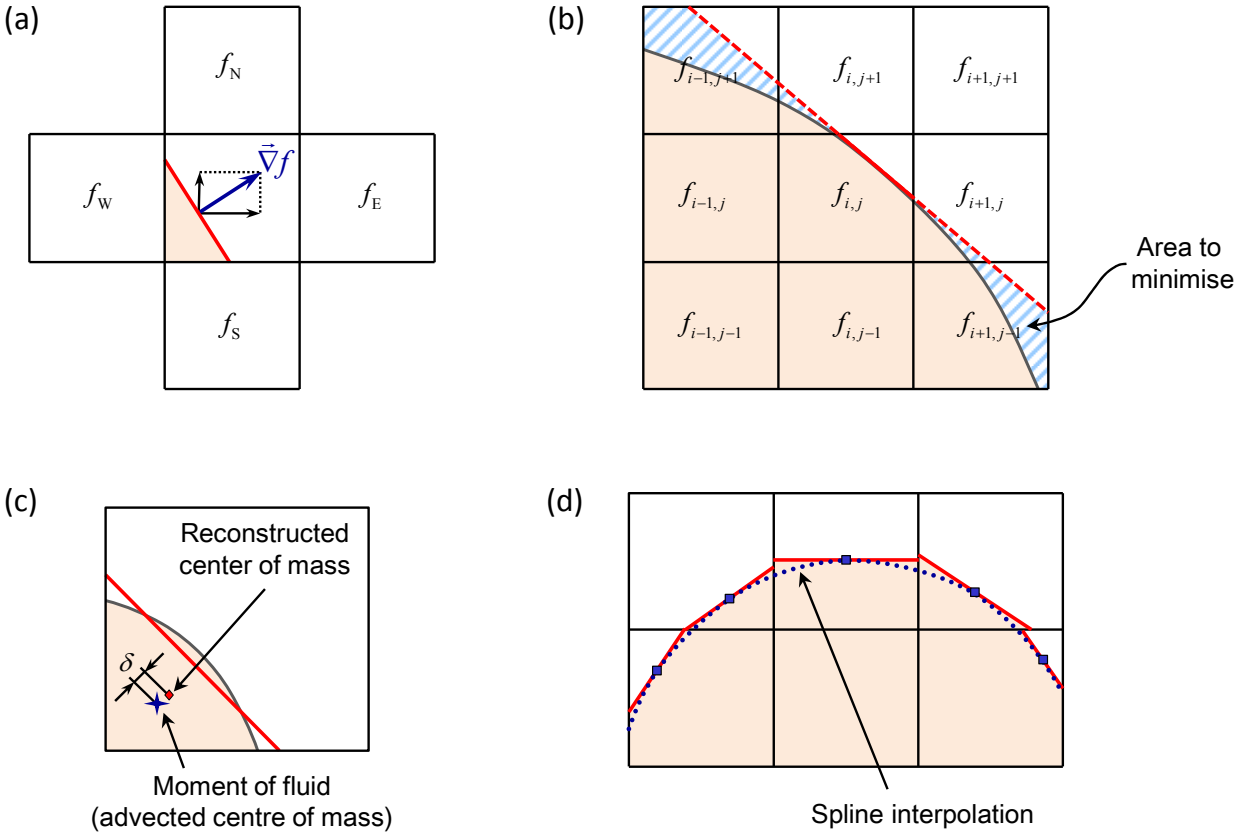


Figure 5.2: Various techniques for the piecewise linear interface calculation: (a) gradient method, (b) least-square minimization error, (c) moment-of-fluid method, (d) spline interpolation correction.

An alternative method, the moment-of-fluid interface reconstruction algorithm (Dyadechko & Shashkov 2006), evaluates the interface normal without information of the liquid volume fraction in the neighboring cells. Instead, it uses the position of the center of mass of the liquid phase in the cell as additional information. The center of mass of the liquid phase in each cell is also advected with the flow, similarly as Lagrangian markers.

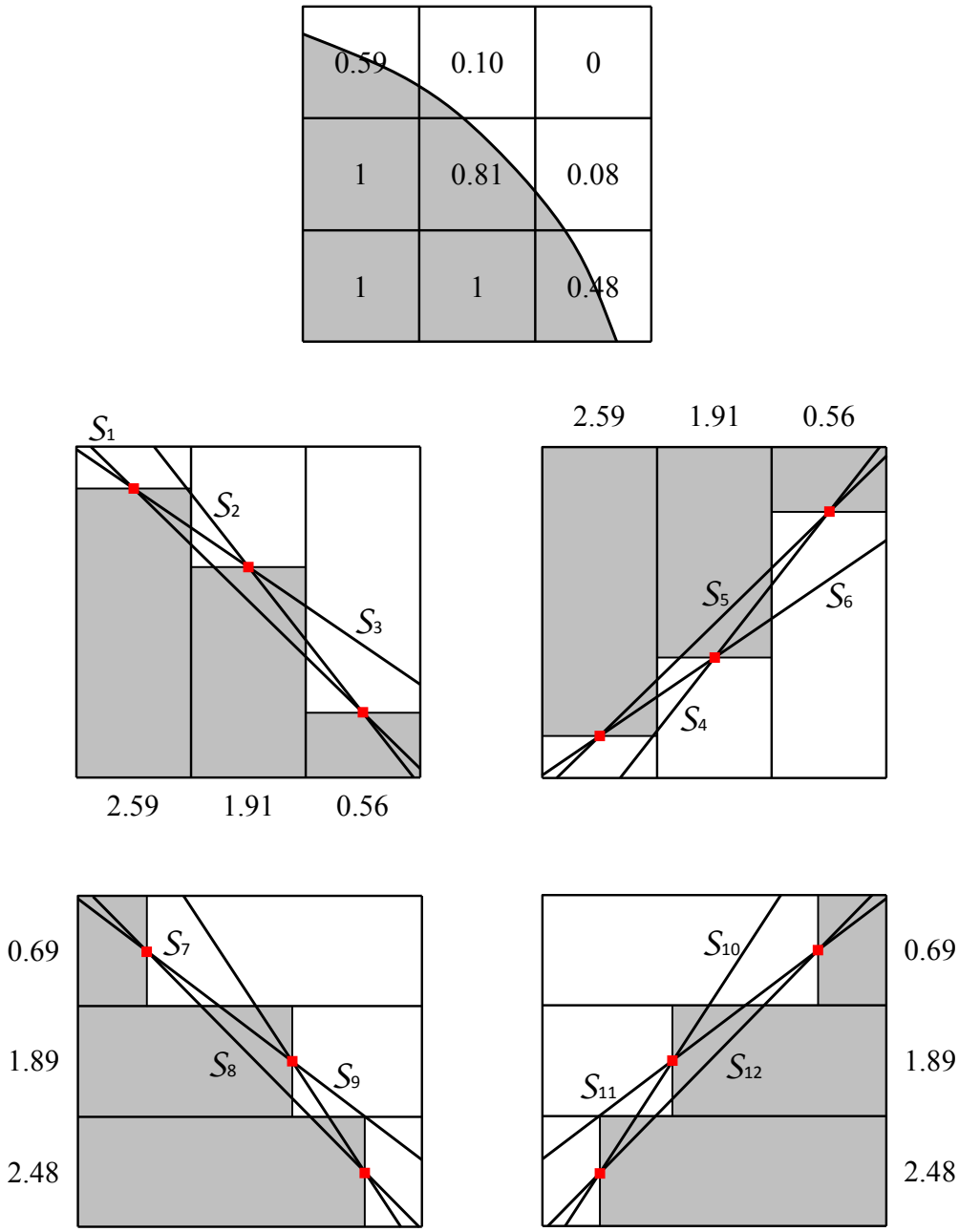


Figure 5.3: The 12 different slope candidates calculated with various finite difference schemes from the column and row sums of the liquid volume in the 3×3 stencil of the nearest neighbors.

The interface reconstruction algorithm consists in minimizing the distance δ between the reconstructed center of mass and the advected center of mass; see Figure 5.2(c). The moment-of-fluid algorithm produces notable improvement in the reconstructions of sharp corners. Finally, smoothening the variations in the curvature can enhance the accuracy of the interface reconstruction; see Figure 5.2(d). The orientation of the PLIC is corrected based on spline interpolations of the slope along the interface (Ginzburg & Wittum 2001; López et al. 2004). This method also provides a better estimation of the surface curvature, for the modeling of surface tension.

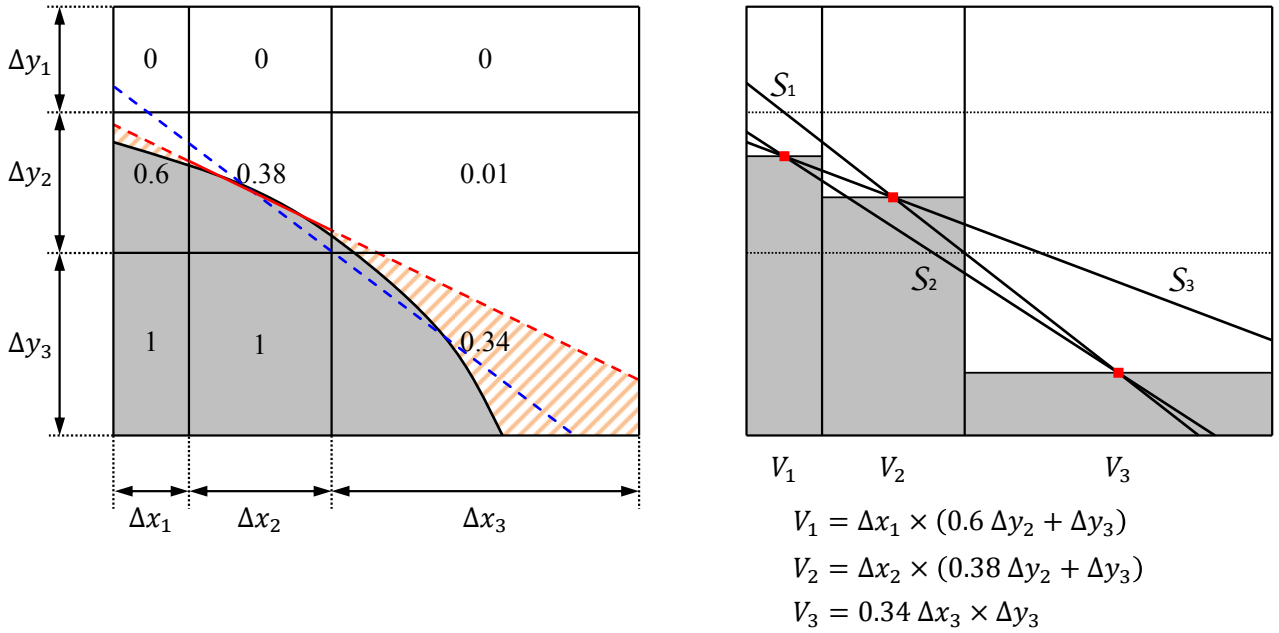


Figure 5.4: ELVIRA interface reconstruction on a non-homogeneous grid. Best fitting interface in red;

In the following, we use an efficient variant of the original least-square method, the ELVIRA method (Pilliod Jr & Puckett 2004), where the algorithm chose the best candidate that minimized the least square error from 12 different interface normals. The different interface normal candidates are calculated with various backward, central and forward finite differences of the row and column sums of the liquid volume fraction in the 3×3 stencil of the nearest neighbors, see Figure 5.3. The ELVIRA is also second-order accurate and it reconstructs all line interfaces exactly. Moreover, it produces very similar results to the LVIRA method.

5.3. Advection schemes

5.3.1. Directional-split advection scheme

Once the piecewise linear interfaces are reconstructed, the new liquid volume fractions are updated using a geometric advection algorithm, which evaluates the fluxed liquid volume from the geometric determination of the DR. The first generations of VOF algorithms were based on directional-split advection schemes, where the liquid volume is transported along one direction at a time. There are two approaches to propagate the liquid volume: the backward *Eulerian implicit* method, and the forward *Lagrangian explicit* method (Zhang 2013). Many variants of the directional-split advection scheme have been presented in the literature.

In the directional-split scheme, the unidimensional donating regions of the fluxes are calculated as rectangles, see Figure 5.5. The liquid volume is only advected in one direction at a time. Hence, the liquid volume VOF algorithm advects the liquid in one direction, then the interface is reconstructed, and finally the fluid is advected in the other direction. The directional-split advection schemes typically use a backward Eulerian method for the time-integration. At each time-step, the first and second directions of advections are switched in order to remove any directional preference of the algorithm. It must be note that since the flow is two-dimensional, the volume is not conserved during the sub-steps of the directional-split advection, since the projection of the two-dimensional

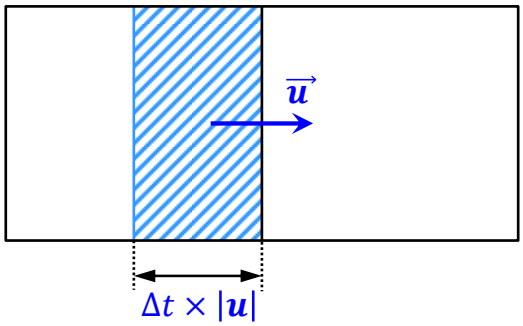


Figure 5.5: Rectangular donating region in the directional-split advection scheme.

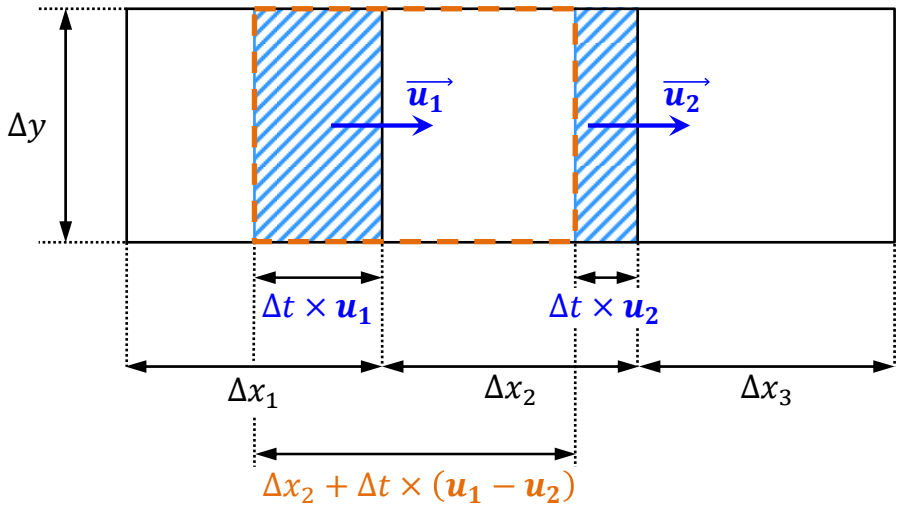


Figure 5.6: Non-conservation of the volume in the sub-steps of the directional-split scheme.

incompressible velocity field in one direction can be compressive or dilating, see Figure 5.6. Thus, the intermediate liquid volume fractions must be calculated with respect to the intermediate volume inside the cell (represented in orange in the Figure 5.6), which differs from the cell's volume. The non-conservation of the volume in the first sub-step of the direction-split advection is compensated by the second sub-step; however, the liquid volume is not rigorously conserved.

Only two directional-split methods conserve the liquid volume exactly: the combined Eulerian Implicit–Lagrangian Explicit (EI-LE) method (Scardovelli & Zaleski 2003; Aulisa et al. 2003; Aulisa et al. 2007), and the Eulerian implicit method of Weymouth and Yue (Weymouth & Yue 2010), which properly integrates the 1D dilatation terms (i.e. the stretching of the velocity field). Nonetheless, all the directional-split advection schemes suffer from numerical diffusion. The lack of accuracy generally produces geometrical errors that distort the interface (Pilliod Jr & Puckett 2004), particularly in non-uniform flows such as in simple shear. In addition, non-conservative schemes also produce unphysical liquid volume fractions (outside the interval [0, 1]), requiring the use of local volume redistribution algorithms to remove the overshoots ($f > 1$) and undershoots ($f < 0$). Moreover, the directional-split methods require intermediate interface reconstructions between each 1D advection, and the Courant number is often limited to maximum 0.5. In contrast, unsplit (or multidimensional) schemes advect the liquid volume in all the directions at once, and do not require intermediate interface reconstructions.

5.3.2. Cellwise conservative unsplit advection scheme

Unsplit advection schemes have been developed to improve accuracy and overcome numerical artifacts of the directional-split schemes. In addition, unsplit schemes are easily generalizable to unstructured mesh. The classical edgewise solution of the VOF advection—based on the flux balance—reads (Zhang 2013):

$$f_C\left(t^{n+1}\right) = f_C\left(t^n\right) + \frac{1}{V_C}\sum_{\text{edge}}\left\|\mathcal{D}_e^- \cap \mathcal{M}\left(t^n\right)\right\| - \left\|\mathcal{D}_e^+ \cap \mathcal{M}\left(t^n\right)\right\|, \tag{5.6}$$

where V_C is the total volume of the cell C, $\mathcal{M}\left(t^n\right)$ is the volume occupied by the liquid phase at the time t^n, $\mathcal{D}_e^-$ and $\mathcal{D}_e^-$ are the received and donated volumes, respectively, through the edge e during the time increment $\Delta t = t^{n+1} - t^n$. Since the flow map is a homeomorphism, the edgewise solution (5.6) is equivalent to the following cellwise solution:

$$f_C\left(t^{n+1}\right) = \frac{1}{V_C}\left\|\overleftarrow{\mathcal{P}_{\Delta t}(C)} \cap \mathcal{M}\left(t^n\right)\right\|, \tag{5.7}$$

where $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is the pre-image of C, traced backward within the time increment Δt.

The cellwise conservative unsplit (CCU) advection scheme computes the new liquid volume fraction $f_C\left(t^{n+1}\right)$ from the formula (5.7). The volume $\mathcal{M}\left(t^n\right)$ occupied by the liquid phase is calculated with the interface reconstruction algorithm. The main task of the CCU scheme is the determination of the pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ of the cell. It is performed in two steps:

i. First, the positions of the cell's vertices are traced backward in time with a 4th-order accurate Runge-Kutta integration. The intermediate velocities are evaluated with bicubic spline interpolations in space, and quadratic spline interpolations in time. The pre-images of the four cell vertices form a polygon, which is a first approximation of the cell's pre-image, yet non-conservative.
ii. Four additional vertices are added to the definition of the pre-image polygon, in order to adjust its volume without changing the position of four vertices obtained via backward tracing. The geometric volume flux correction uses explicit formulas—rather than iterations—based on the DR calculation.

Finally, the new liquid volume fraction is obtained from the intersection of the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ with the reconstructed PLIC. Each of these steps is described hereinafter.

5.3.2.1. Backward-tracing

The accuracy of the CCU scheme lies in the backward tracing of the cell's pre-image. We consider a rectangular grid cell C defined by its vertices 1,2,3,4. Vertices A,B,C,D are obtained from the backward tracing of 1,2,3,4, respectively. The polygon ABCD is the first approximation of the cell's pre-image, as shown in Figure 5.7(a).

The backward tracing of a vertex is obtained with the reversed time-integration of the instantaneous velocity along the pathline of a Lagrangian particle p, initially located at the vertex's position. The instantaneous velocity of the particle p whose position $\boldsymbol{x}_p(t)$ varies in time is

$$\boldsymbol{v}(p,t) = \frac{d\boldsymbol{x}_p}{dt}. \tag{5.8}$$

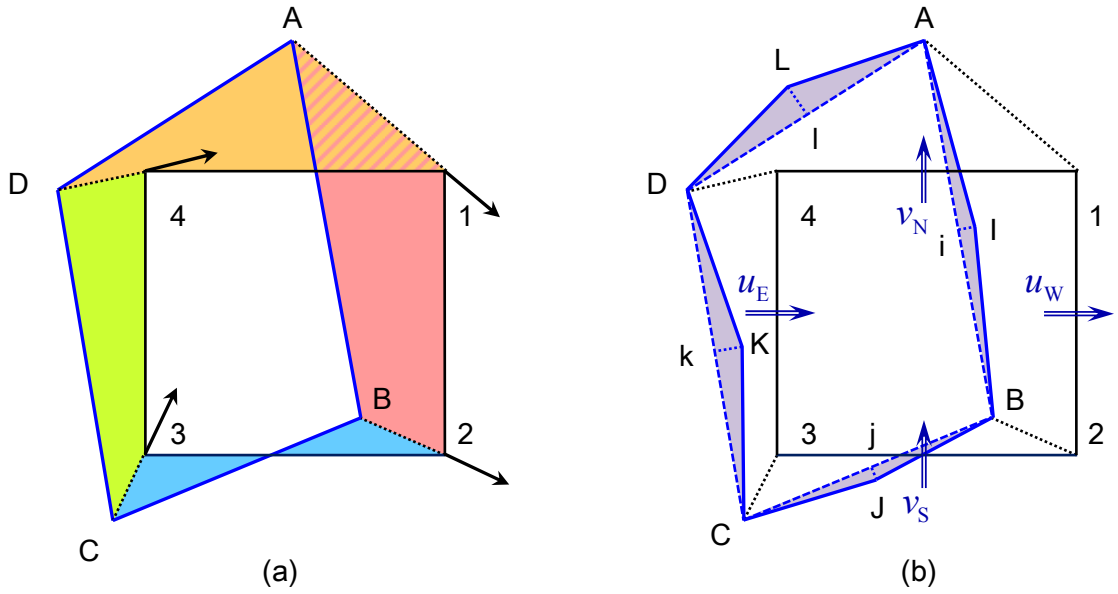


Figure 5.7: Calculation of the pre-image polygon: (a) non-conservative 4-vertex approximation via backward tracing of the cell's vertex, (b) conservative 8-vertex polygon after the geometric volume flux correction

The pre-image of the vertex is

$$\boldsymbol{x}_p\left(t^n\right)=\boldsymbol{x}_p\left(t^{n+1}\right)-\int_{t^n}^{t^{n+1}}\boldsymbol{v}\left(p,t^*\right)\mathrm{d}t^* \tag{5.9}$$

In our scheme, the integral in (5.9) is solved numerically with the explicit 3/8 forth-order Runge-Kutta method:

$$\begin{aligned}
\boldsymbol{x}_p\left(t^n\right)&=\boldsymbol{x}_p\left(t^{n+1}\right)-\frac{\Delta t}{8}\left(\boldsymbol{v}_1+3\boldsymbol{v}_2+3\boldsymbol{v}_3+\boldsymbol{v}_4\right)+\underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^5\right),\\
\boldsymbol{v}_1&=\boldsymbol{u}\left(t^{n+1},\boldsymbol{x}_p\left(t^{n+1}\right)\right),\\
\boldsymbol{v}_2&=\boldsymbol{u}\left(t^{n+2/3},\boldsymbol{x}_p\left(t^{n+1}\right)-\Delta t\,\boldsymbol{v}_1/3\right),\\
\boldsymbol{v}_3&=\boldsymbol{u}\left(t^{n+1/3},\boldsymbol{x}_p\left(t^{n+1}\right)+\Delta t\left(\boldsymbol{v}_1/3-\boldsymbol{v}_2\right)\right),\\
\boldsymbol{v}_4&=\boldsymbol{u}\left(t^{n},\boldsymbol{x}_p\left(t^{n+1}\right)-\Delta t\left(\boldsymbol{v}_1-\boldsymbol{v}_2+\boldsymbol{v}_3\right)\right),
\end{aligned} \tag{5.10}$$

where $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$, $\boldsymbol{v}_4$ are intermediate velocities of the particle, and $\boldsymbol{u}\left(t,\boldsymbol{x}\right)$ is the Eulerian velocity field. The VOF method is typically coupled with an implicit (or semi-implicit) Navier-Stokes solver, which resolves $\boldsymbol{u}\left(t,\boldsymbol{x}\right)$ on a discrete set of locations, at discrete points in time. The values of the intermediate velocities are obtained from the temporal and spatial interpolations of the known discrete velocities.

In order to avoid velocity overshoots and to promote numerical stability, it is preferable to first solve the new velocity field implicitly, prior to solving the new liquid volume fraction explicitly. The velocities at the intermediate times $t^{n+1/3}$ and $t^{n+2/3}$ are determined by quadratic spline interpolation within the time interval $t\in\left[t^n,t^{n+1}\right]$:

$$\boldsymbol{u}(t)=\boldsymbol{u}\left(t^n\right)+\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\left(t-t^n\right)+\left[\boldsymbol{u}\left(t^{n+1}\right)-\boldsymbol{u}\left(t^n\right)-\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\Delta t\right]\left(\frac{t-t^n}{\Delta t}\right)^2, \tag{5.11}$$

where the acceleration $\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}$ is known from the previous time-step. The acceleration at the end of the current time-step is evaluated and saved for use at the next time-step:

$$\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^{n+1}}=\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}+2\left[\frac{\boldsymbol{u}\left(t^{n+1}\right)-\boldsymbol{u}\left(t^n\right)}{\Delta t}-\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\right]. \tag{5.12}$$

The temporal quadratic approximation for the evolution of the velocity provides a balance between accuracy and efficiency. In our scheme, the accuracy of the backward tracing is enhanced by the use of bicubic spatial interpolations for the intermediate velocities:

$$\boldsymbol{u}(\xi,\eta)=\sum_{i=0}^{3}\sum_{j=0}^{3}a_{ij}\xi^i\eta^j, \tag{5.13}$$

where ξ and η are the normalized local coordinates inside the cell, and a_{ij} are the 16 coefficients of the different terms.

The bicubic spline interpolation g of a scalar field s (either one of the velocity components u, v, or the streamfunction ϕ) is a polynomial function with 16 terms:

$$\begin{aligned}g(\xi,\eta)=\sum_{i=0}^{3}\sum_{j=0}^{3}a_{ij}\xi^i\eta^j=a_{00}+a_{01}\eta+a_{02}\eta^2+a_{03}\eta^3+a_{10}\xi+a_{11}\xi\eta+a_{12}\xi\eta^2+a_{13}\xi\eta^3+\ldots\\ a_{20}\xi^2+a_{21}\xi^2\eta+a_{22}\xi^2\eta^2+a_{23}\xi^2\eta^3+a_{30}\xi^3+a_{31}\xi^3\eta+a_{32}\xi^3\eta^2+a_{33}\xi^3\eta^3,\end{aligned} \tag{5.14}$$

where $(\xi,\eta)\in[0,1]^2$ are the normalized local coordinates inside the rectangular grid cell, see Figure 5.8. The value of the field *s* is known at the cell's vertices (0,0), (1,0), (1,0), (1,1). The first and cross derivatives s_x, s_y and s_{xy} are further evaluated at these vertices by finite-differences. Fitting these 4 values at the 4 corners provides 16 equations for the coefficients a_{ij}:

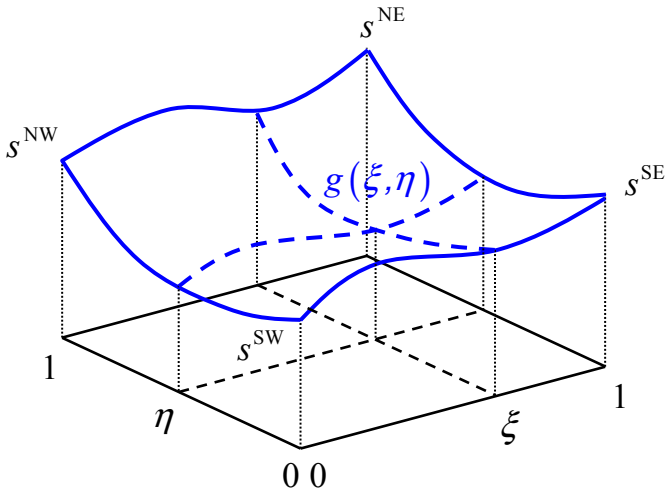


Figure 5.8: Bicubic spline interpolation inside a rectangular grid cell, using normalized local coordinate system.

$$
\begin{aligned}
s^{\mathrm{SW}} &= g(0,0) = a_{00},\\
s^{\mathrm{SE}} &= g(1,0) = a_{00} + a_{10} + a_{20} + a_{30},\\
s^{\mathrm{NW}} &= g(0,1) = a_{00} + a_{01} + a_{02} + a_{03},\\
s^{\mathrm{NE}} &= g(1,1) = a_{00} + a_{01} + a_{02} + a_{03} + a_{10} + a_{11} + a_{12} + a_{13} + \ldots\\
&\qquad a_{20} + a_{21} + a_{22} + a_{23} + a_{30} + a_{31} + a_{32} + a_{33}\ ,
\end{aligned}
\tag{5.15}
$$

$$
\begin{aligned}
\Delta x\ s_x^{\mathrm{SW}} &= \frac{\partial g}{\partial \xi}(0,0) = a_{10},\\
\Delta x\ s_x^{\mathrm{SE}} &= \frac{\partial g}{\partial \xi}(1,0) = a_{10} + 2a_{20} + 3a_{30},\\
\Delta x\ s_x^{\mathrm{NW}} &= \frac{\partial g}{\partial \xi}(0,1) = a_{10} + a_{11} + a_{12} + a_{13},\\
\Delta x\ s_x^{\mathrm{NE}} &= \frac{\partial g}{\partial \xi}(1,1) = a_{10} + a_{11} + a_{12} + a_{13} + 2a_{20} + 2a_{21} + 2a_{22} + \ldots\\
&\qquad 2a_{23} + 3a_{30} + 3a_{31} + 3a_{32} + 3a_{33},
\end{aligned}
\tag{5.16}
$$

$$
\begin{aligned}
\Delta y\ s_y^{\mathrm{SW}} &= \frac{\partial g}{\partial \eta}(0,0) = a_{01},\\
\Delta y\ s_y^{\mathrm{SE}} &= \frac{\partial g}{\partial \eta}(1,0) = a_{01} + a_{11} + a_{21} + a_{31},\\
\Delta y\ s_y^{\mathrm{NW}} &= \frac{\partial g}{\partial \eta}(0,1) = a_{01} + 2a_{02} + 3a_{03},\\
\Delta y\ s_y^{\mathrm{NE}} &= \frac{\partial g}{\partial \eta}(1,1) = a_{01} + 2a_{02} + 3a_{03} + a_{11} + 2a_{12} + 3a_{13} + \ldots\\
&\qquad a_{21} + 2a_{22} + 3a_{23} + a_{31} + 2a_{32} + 3a_{33},
\end{aligned}
\tag{5.17}
$$

$$
\begin{aligned}
\Delta x\Delta y\ s_{xy}^{\mathrm{SW}} &= \frac{\partial^2 g}{\partial \xi \partial \eta}(0,0) = a_{11},\\
\Delta x\Delta y\ s_{xy}^{\mathrm{SE}} &= \frac{\partial^2 g}{\partial \xi \partial \eta}(1,0) = a_{11} + 2a_{21} + 3a_{31},\\
\Delta x\Delta y\ s_{xy}^{\mathrm{NW}} &= \frac{\partial^2 g}{\partial \xi \partial \eta}(0,1) = a_{11} + 2a_{12} + 3a_{13},\\
\Delta x\Delta y\ s_{xy}^{\mathrm{NE}} &= \frac{\partial^2 g}{\partial \xi \partial \eta}(1,1) = a_{11} + 2a_{12} + 3a_{13} + 2a_{21} + 4a_{22} + 6a_{23} + 3a_{31} + 6a_{32} + 9a_{33}.
\end{aligned}
\tag{5.18}
$$

The system of linear relations (5.15)–(5.18) can be expressed in the matrix form

$$
\boldsymbol{A}\alpha = x\ , \tag{5.19}
$$

where

$$
\alpha = \begin{bmatrix} a_{00} & a_{10} & a_{20} & a_{30} & a_{01} & a_{11} & a_{21} & a_{31} & a_{02} & a_{12} & a_{22} & a_{32} & a_{03} & a_{13} & a_{23} & a_{33} \end{bmatrix}^{\mathrm{T}},
$$

$$
x = \begin{bmatrix} s^{\mathrm{SW}} & s^{\mathrm{SE}} & s^{\mathrm{NW}} & s^{\mathrm{NE}} & \Delta x s_x^{\mathrm{SW}} & \Delta x s_x^{\mathrm{SE}} & \Delta x s_x^{\mathrm{NW}} & \Delta x s_x^{\mathrm{NE}} & \Delta y s_y^{\mathrm{SW}} & \Delta y s_y^{\mathrm{SE}} & \Delta y s_y^{\mathrm{NW}} & \Delta y s_y^{\mathrm{NE}} & \ldots \\ \Delta x\Delta y s_{xy}^{\mathrm{SW}} & \Delta x\Delta y s_{xy}^{\mathrm{SE}} & \Delta x\Delta y s_{xy}^{\mathrm{NW}} & \Delta x\Delta y s_{xy}^{\mathrm{NE}} \end{bmatrix}^{\mathrm{T}}.
$$

Finally, the solution for the coefficients a_{ij} is $\alpha = \boldsymbol{A}^{-1}x$, with

$$\boldsymbol{A}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 \\ -3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 \\ 9 & -9 & -9 & 9 & 6 & 3 & -6 & -3 & 6 & -6 & 3 & -3 & 4 & 2 & 2 & 1 \\ -6 & 6 & 6 & -6 & -3 & -3 & 3 & 3 & -4 & 4 & -2 & 2 & -2 & -2 & -1 & -1 \\ 2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ -6 & 6 & 6 & -6 & -4 & -2 & 4 & 2 & -3 & 3 & -3 & 3 & -2 & -1 & -2 & -1 \\ 4 & -4 & -4 & 4 & 2 & 2 & -2 & -2 & 2 & -2 & 2 & -2 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{5.20}$$

As a result, the interpolated velocity field is C^1-continuous and varies smoothly in both time and space. In addition, the quadratic temporal and bicubic spatial interpolations allow the velocity field to have local extrema in between the known discrete values.

Finally the positions of the vertices A,B,C,D are determined using formulas (5.10)–(5.13). The resulting polygon ABCD is an overall good approximation of the cell's pre-image, yet it does not fulfill the volume conservation (its volume is not strictly equal to the volume of C).

5.3.2.2. Geometric volume flux correction

The volume conservation of the cell is enforced by adjusting the area of its pre-image polygon. The positions of the vertices A,B,C,D, computed with overall fourth-order accuracy, are kept unchanged in order to have conforming adjacent DRs. Instead, four additional vertices I,J,K,L are introduced to the definition of the pre-image polygon, see Figure 5.7(b). They are initially positioned at the midpoints i,j,k,l of the pre-image's edges [AB], [BC], [CD] and [DA], respectively. The area of the pre-image polygon AIBJCKDL is modified by adjusting the position of the vertices I,J,K,L along the median line of the pre-image's edges. Therefore, the volume corrections are added or removed to the pre-image's edges as isosceles triangles. The suitable positions of the vertices I,J,K,L are calculated via the geometric volume flux correction procedure.

The volume conservation is a consequence of the incompressibility constraint, which is the kinematic condition of solenoidal velocity fields. Integrating the divergence of a solenoidal velocity field over the control volume C and applying the divergence theorem yields:

$$\oiint_{\partial C} (\boldsymbol{u} \cdot \hat{\boldsymbol{n}}) \, \mathrm{d}s = 0 \,. \tag{5.21}$$

It translates into the continuity equation:

$$\sum_e A_e u_e = 0 \,, \tag{5.22}$$

where e is the index of the edges, A_e are the lengths of the edges, and u_e the face-averaged velocities. The quantity

$$\boldsymbol{\Phi}_e = A_e u_e \tag{5.23}$$

is the volume flux through the edge e. By virtue of the equivalence between the cellwise and edgewise approaches, an easy and robust way to enforce the volume conservation on the pre-image is to make the net volume fluxes of the DRs equal to the divergence-free volume fluxes $\boldsymbol{\Phi}_e$.

The calculation of the DR requires the evaluation of the *timelines* of the cell's faces, and the *streaklines* of the cell's vertices, see Figure 5.9. Note that in transient flows, the streaklines are different from the pathlines

calculated in the backward tracing, yet they have the same extremities. A streakline is the trace of particles continuously injected at a certain location, while a pathline is the trajectory of an individual particle. In our scheme, two discrete points 1' and 1'' of the streaklines are calculated via backward tracing of particles injected in the flow at the cell's vertices, at the intermediate times $t^{n+1/3}$ and $t^{n+2/3}$. In order to avoid further interpolations of the velocity field, the position of the point 1' is obtained by the two-level Heun's method:

$$\begin{aligned} \boldsymbol{x}_p\left(t^n\right) &= \boldsymbol{x}_p\left(t^{n+1/3}\right) - \frac{\Delta t}{6}\left(\boldsymbol{v}_1 + \boldsymbol{v}_2\right) + \underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^3\right), \\ \boldsymbol{v}_1 &= \boldsymbol{u}\left(t^{n+1/3}, \boldsymbol{x}_p\left(t^{n+1/3}\right)\right), \\ \boldsymbol{v}_2 &= \boldsymbol{u}\left(t^{n}, \boldsymbol{x}_p\left(t^{n+1/3}\right) - \Delta t\boldsymbol{v}_1/3\right), \end{aligned} \tag{5.24}$$

and the position of the point 1'' is obtained by the Kutta's three-level method:

$$\begin{aligned} \boldsymbol{x}_p\left(t^n\right) &= \boldsymbol{x}_p\left(t^{n+2/3}\right) - \frac{\Delta t}{9}\left(\boldsymbol{v}_1 + 4\boldsymbol{v}_2 + \boldsymbol{v}_3\right) + \underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^4\right), \\ \boldsymbol{v}_1 &= \boldsymbol{u}\left(t^{n+2/3}, \boldsymbol{x}_p\left(t^{n+2/3}\right)\right), \\ \boldsymbol{v}_2 &= \boldsymbol{u}\left(t^{n+1/3}, \boldsymbol{x}_p\left(t^{n+2/3}\right) - \Delta t\boldsymbol{v}_1/3\right), \\ \boldsymbol{v}_3 &= \boldsymbol{u}\left(t^{n}, \boldsymbol{x}_p\left(t^{n+2/3}\right) + \Delta t\left(2\boldsymbol{v}_1 - 4\boldsymbol{v}_2\right)/3\right). \end{aligned} \tag{5.25}$$

Note that the variable Δt in the formulas (5.24) and (5.25) is the time increment of the whole time-step, not the time interval of the Heun or the Kutta's methods. Once the points 1' and 1'' are determined, the DR is represented by the polygon 11'1''AiB2''2'2.

The error in the calculation of the DR has two contributions. The first contribution comes from the integration errors in the backward tracing using the formulas (5.10), (5.24) and (5.25), i.e the calculated positions of points 1',1'',A are different from their theoretical positions $\tilde{1}'$,$\tilde{1}''$,$\tilde{A}$. The second error contribution is due to the reconstructed representation of the streaklines and timelines. For higher precision, the streaklines are represented in our scheme with polygonal chains 1–1'–1''–A. The use of three sub-segments provides a balance between accuracy and efficiency.

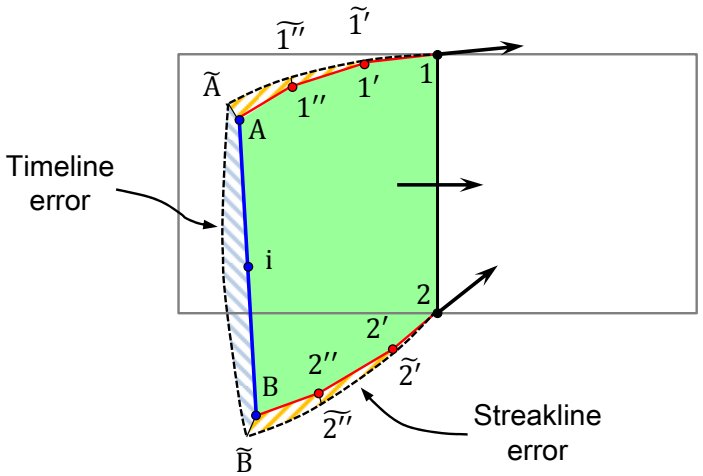


Figure 5.9: Calculation of the DR of the edge 12. The polygonal chains 1–1'–1''–A and 2–2'–2''–B, and the segments A–i–B are the approximate representations of the streaklines and timeline, whose exact positions are represented by the black dashed curves. The hatched areas represent geometrical errors of the DR construction.

The net volume flux F_e through a cell's edge e is calculated via the area of its DR polygon. For a regular—non-self-intersecting—DR polygon $\mathcal{P}_e$, depicted in Figure 5.10(a), the net volume flux is simply its area, computed with the following formula:

$$F_e = \|\mathcal{P}_e\| = \frac{1}{2}\left|\sum_{i=0}^{n-1} x_i y_{i+1} - x_{i+1} y_i\right|, \tag{5.26}$$

where n is the number of vertices defining the polygon $\mathcal{P}_e$, and (x_i, y_i) are their coordinates. The formula (5.26) is valid for both convex and concave polygons. Furthermore, particular attention must be paid to the case of irregular—self-intersecting—polygons. We distinguish between two cases of self-intersections in the DR polygons, based on whether they involve the cell's edges or not. The CCU scheme only allows self-intersections involving the cell's edges. The ability of an advection scheme to properly deal with this event is crucial for its accuracy. Directional-split are not able to take into account both entering and exiting volumes that may cross an edge within the same time increment. The CCU scheme differentiates the entering and exiting volumes of an edge e, by splitting the admissible irregular DR polygon $\mathcal{P}_e$ into two entities $\mathcal{D}_e^+$ and $\mathcal{D}_e^-$, whose contributions to the net volume flux have opposite signs, such that

$$\begin{aligned} \mathcal{D}_e^+ \cup \mathcal{D}_e^- &= \mathcal{P}_e, \\ \mathcal{D}_e^+ \cap \mathcal{D}_e^- &\subset e. \end{aligned} \tag{5.27}$$

The areas of $\mathcal{D}_e^+$ and $\mathcal{D}_e^-$ are the donated and received volumes of the cell C through the edge e, respectively, see Figure 5.10(b). Then, the net volume flux of an admissible irregular DR through the edge e is given by:

$$F_e = \|\mathcal{D}_e^-\| - \|\mathcal{D}_e^+\|. \tag{5.28}$$

Finally, the amount of volume flux correction that the DR requires is given by the difference

$$\Delta\phi_e = \Phi_e - F_e. \tag{5.29}$$

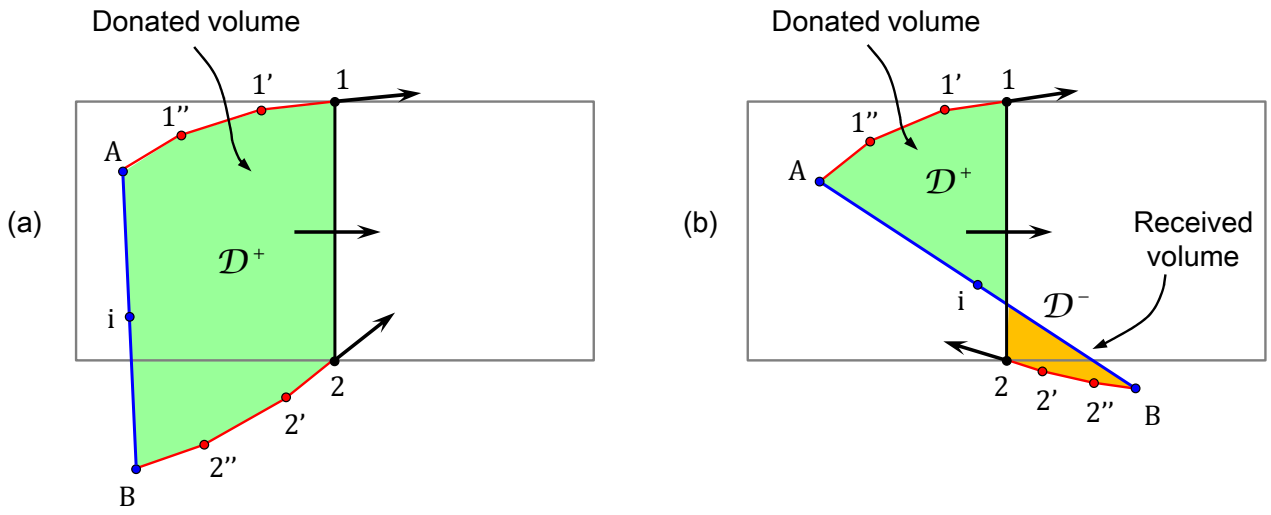


Figure 5.10: Calculation of the volume flux through the edge 12, (a) in the case of a regular DR polygon, (b) in the case of an admissible self-intersecting DR polygon.

The needed volume flux correction is added or removed to the area of the DR by moving the position of the vertex I from the midpoint i of the segment [AB], by the distance h_e along the bisector line of [AB], enforcing systematic conservativeness, see Figure 5.7(b). The volume flux corrections are the isosceles triangles colored in purple, thus

$$h_e = 2\Delta\phi_e / \|\mathrm{AB}\|. \tag{5.30}$$

The algorithm is valid for all the topologies of admissible DRs: convex, concave and self-intersecting, the three of them being represented in Figure 5.7(b). After application of the volume flux correction to all the DRs, the resulting 8-vertex polygon AIBJCKDL is a conservative representation of the cell's pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$.

5.3.2.3. Cellwise VOF update

The last step of the cellwise advection consists in computing the intersection of the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ with the area $\mathcal{M}(t^n)$ occupied by the liquid material. $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is determined by the CCU scheme while $\mathcal{M}(t^n)$ is obtained from the PLIC interface reconstruction. The intersection of polygons is a Boolean operation that requires algorithms of *combinatorial computational geometry*. The intersection $\overleftarrow{\mathcal{P}_{\Delta t}(C)} \cap \mathcal{M}(t^n)$ is computed using a simple line/line intersection algorithm. A search algorithm is first used to find all the underlying grid cells with potential intersection with $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$. These underlying cells have at least one of their vertices within the interval $\left[\min\left(x_{\overleftarrow{\mathcal{P}(C)}}\right), \max\left(x_{\overleftarrow{\mathcal{P}(C)}}\right)\right] \times \left[\min\left(y_{\overleftarrow{\mathcal{P}(C)}}\right), \max\left(y_{\overleftarrow{\mathcal{P}(C)}}\right)\right]$, bounded by the minimum and maximum values of the coordinates of the pre-image polygon's vertices. Then, the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is split into its intersections with each of the underlying grid cells, see Figure 5.11. The different cuts of $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ are obtained using the TRUNCATEPOLYGONBYLINE algorithm, described below. The liquid volume inside each cut of the pre-image polygon is obtained by further truncation with the PLIC line. Finally, the cellwise update of the liquid volume fraction is recovered by summing up the liquid volume contributions from all the truncations of the pre-image polygon, corresponding to the area colored in purple in Figure 5.11.

5.3.3. Numerical tests

5.3.3.1. Pre-image calculation test

In this subsection, the accuracy of the CCU pre-image calculation is tested in benchmark flow fields, in order to differentiate between pure advection errors and interface reconstruction errors. The pre-images construction error E_1 is defined as the area of the difference (exclusive OR) between the computed 8-vertex pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{CCU}}$ and the exact pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{exact}}$, summed for all the grid cells C:

$$E_1 \equiv \sum_C \left\| \overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{CCU}} \oplus \overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{exact}} \right\|. \tag{5.31}$$

In the numerical tests, the "exact" pre-image of each cell is a very fine polygonal representation with 80 vertices, whose positions are obtained by numerical integration with a predictor-corrector scheme over 20 sub-divisions of the time-step.

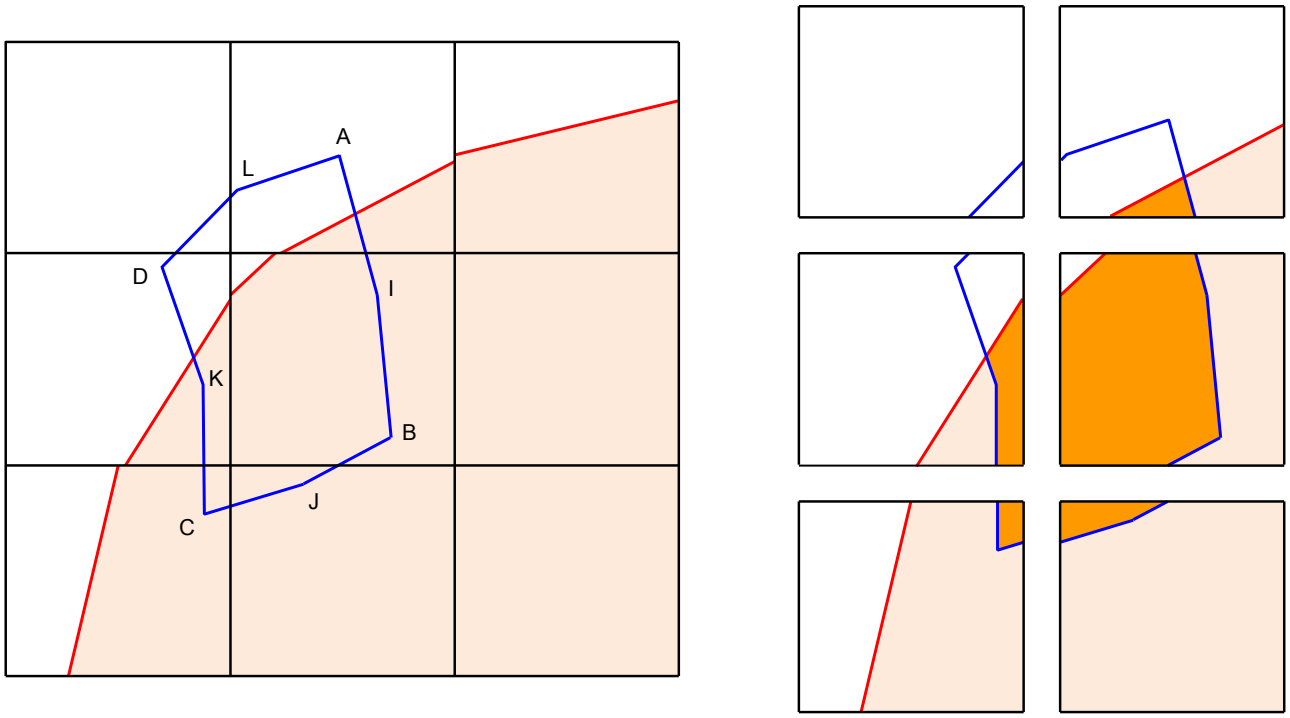


Figure 5.11: Truncation of the pre-image polygon AIBJCKDL with the underlying grid cells and their reconstructed PLIC lines. The liquid material occupies the colored area. The dark area represents the final intersection of the pre-image with the liquid material.

Algorithm TRUNCATEPOLYGONBYLINE: Compute the truncation of a polygon by an oriented line.

```
1:  function TruncatePolygonByLine(Q, n̂, d)
2:      input Q                                   List of the polygon's vertices {Q1, Q2...}
3:      input n̂                                   Normal vector of the oriented line
4:      input d                                   Distance to origin of the oriented line
5:      if n̂ · Q1 + d > 0 then in1 ← false        The first polygon's vertex is outside the cut
6:      else in1 ← true end if                    The first polygon's vertex belongs to the cut
7:      if in1 then P ← {Q1}                      Save the first polygon's vertex in the output list
8:      else P ← {∅} end if                       Create an empty output list
9:      for k = 2 → NumberOfElements(Q) do        Loop over the polygon's edges
10:         if n̂ · Qk + d > 0 then in2 ← false    The k^th polygon's vertex is outside the cut
11:         else in2 ← true end if                The k^th polygon's vertex belongs to the cut
12:         if in1 XOR in2 then                   Evaluate if the edge crosses the oriented line
13:             n̂2 ← (Qk − Qk−1)^⊥ / ‖Qk − Qk−1‖   Calculate the normal vector of the edge
14:             d2 ← Qk · n̂2                      Calculate the distance to origin of the edge
15:             s ← ComputeLinesIntersection(n̂, d, n̂2, d2)   Compute the intersection of the two lines
16:             if s ∈ [Qk Qk-1] AND (s ≠ Qk AND s ≠ Qk-1) then
17:                 P ← {P, s} end if             Add the intersection point to the output list
18:         end if
19:         if in2 then P ← {P, Qk} end if        Add the k^th polygon's vertex to the output list
20:         in1 ← in2
21:     end for
22:     return P                                  Return the vertex list of the truncated polygon
23: end function
```

The pre-image calculation error E_1 is very relevant to assess the accuracy of cellwise advection schemes, as the liquid volume fraction is obtained from the intersection of the interface with the area of whole the cell's pre-image. In fact, E_1 is an upper bound of the total advection error, independent of the interface position and the interface reconstruction, i.e. if all the cells were affected by the update of the liquid volume fraction and if the interface was represented exactly. Moreover, E_1 is a norm since it is zero when the pre-images are exactly constructed. The pre-image calculation error produced by the CCU scheme is measured for various incompressible flows within the rectangular calculation domain $[0,1]\times[0,1]$. In the tests, the velocities at the cell's vertices are derived from analytical expressions of the streamfunction ϕ, with $u = \mathrm{d}\phi/\mathrm{d}y$ and $v = -\mathrm{d}\phi/\mathrm{d}x$. The face-averaged velocities are obtained by the difference in the streamfunction level between extremities of the edges. The following flows are considered:

- the simple translation field: $\phi(x,y) = x + y$,
- the simple rotation field: $\phi(x,y) = (x-0.5)^2 + (y-0.5)^2$,
- the single vortex in-a-box field: $\phi(x,y) = -\frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)$,
- the 4-vortex deformation field: $\phi(x,y) = \frac{1}{4\pi}\sin\left(4\pi(x+0.5)\right)\cos\left(4\pi(y+0.5)\right)$.

These velocity fields have symmetries and periodicities. Figure 5.12 shows the computed versus "exact" pre-images, as well as the streaklines of the grid vertices. Note that Figure 5.12 only displays representative unit cells of the flows, for relatively coarse grids. The reconstructed DRs and pre-images do not overlap and are conforming to the edge of their neighbors (consistency of the streaklines). The pre-image calculation errors reported in Table 5.1 are computed for the entire domain, during one time-step, using the maximum Courant number $C_{\max} = 1$.

In simple translation, the grid's pre-image is constructed exactly (to the machine precision), for all grid sizes. The increase in the value of E_1 with the mesh refinement is due to the accumulation of round-off errors. In the simple rotation, very small errors are generated. Careful examination of the calculated pre-image reveals that the pre-images of the grid vertices—traced backward—are determined exactly, because the rotational velocity field is linear in terms of space. The small errors come from the approximation of the streaklines as polygonal chains, while the exact streaklines are circle arcs. Errors in the area of the DR results in positioning errors of the mid-edge correcting vertex of the volume flux correction. The error E_1 is reduced with mesh refinements, with a convergence rate between 4 and 5, and it reaches the precision of accumulated round-off errors for the mesh 128×128.

In the single vortex in-a-box and the 4-vortex deformation fields, additional errors come from the backward tracing since the bicubic spatial interpolation of the velocity does not represent sine and cosine functions exactly. However, considering the coarseness of the mesh, these errors are considered relatively small. Bicubic splines have accuracy of order $O(\Delta x^4)$, but the number of cells increase as $O(1/\Delta x^2)$ with the mesh refinement, hence a formal second-order accuracy. However, the apparent convergence rate of the error is between 2 and 3.

In principle, the error in the simple rotation—only due to approximations of the streaklines with polygonal chains—would vanish if one uses a spline representation of the streaklines, as proposed by Zhang (2013a; 2013b). However, the advection errors in the simple rotation tests, solely due to the streakline representation, are five orders of magnitude smaller than in the single vortex in-a-box tests, where backward tracing also contributes to the errors. The simple rotation tests also have higher convergence rates than the single vortex in-a-box tests. Therefore, the overall accuracy of advection is limited by the accuracy of the backward tracing, i.e. the Runge-Kutta method and the bicubic spatial interpolation, rather than the polygonal chain approximation.

Grid size	8×8	16×16	32×32	64×64	128×128
Simple translation	2.16 e-15	3.22 e-15	1.72 e-14	6.26 e-14	2.58 e-13
Simple rotation	8.68 e-9	3.34 e-10 (4.70)	1.39 e-11 (4.59)	7.92 e-13 (4.13)	1.54 e-13
Single vortex in-a-box	7.00 e-4	9.70 e-5 (2.85)	1.39 e-5 (2.81)	2.10 e-6 (2.73)	3.10 e-7 (2.75)
4-vortex deformation field	-	1.19 e-3	2.72 e-4 (2.13)	5.18 e-5 (2.39)	8.18 e-6 (2.67)

Table 5.1: Pre-image calculation errors E_1 computed for the entire grid, with $C_{\max} = 1$. The convergence rates are in parentheses.

5.3.3.2. Single vortex deformation test

In this subsection, the accuracy of the advection schemes is tested in conjunction with the PLIC–ELVIRA interface reconstruction algorithm. The time-reversed single vortex deformation test is a classical benchmark test. The calculation domain is a rectangle $[0,1]\times[0,1]$. The liquid phase is initially a disk of radius 0.15 and centered at the position $(0.5,\ 0.75)$. The interface of the material deforms during the time interval $t=[0,T]$, by the single vortex in-a-box velocity field:

$$\phi(x,y,t) = -\frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)\cos\left(\frac{\pi t}{T}\right). \tag{5.32}$$

The corresponding velocity field ($u = \partial\phi/\partial y$ and $v = -\partial\phi/\partial x$) is represented in Figure 5.13. The magnitude of the vortex is reversed at the time $t = T/2$; thus the deformed material theoretically recovers its initial shape at the final time T. The test is performed for the vortex period $T = 8$, with different grid spacing of the Cartesian mesh.

For the directional-split advection scheme, the error in the volume conservation is measured as:

$$E_c(n) = \sum_{i,j}^{n} A_{i,j} f_{i,j}^{T} - \sum_{i,j}^{n} A_{i,j} f_{i,j}^{0} \tag{5.33}$$

For the CCU advection scheme, the volume is rigorously conserves to the machine precision, because of the volume flux correction. Geometrical errors (in the position of the liquid volume) are measured as the 1-norm of the difference between the initial and final liquid volume fraction inside all the cells:

$$E_g(n) = \sum_{i,j}^{n} A_{i,j}\left|f_{i,j}^{T} - f_{i,j}^{0}\right|, \tag{5.34}$$

where $A_{i,j}$ is the area of the i,j^{th} grid cell, and n is the number of grids along each dimension. The convergence rate of the error with mesh refinement is given by:

$$O(n) = \frac{\ln\left(E_g(2n)/E_g(n)\right)}{\ln(1/2)}. \tag{5.35}$$

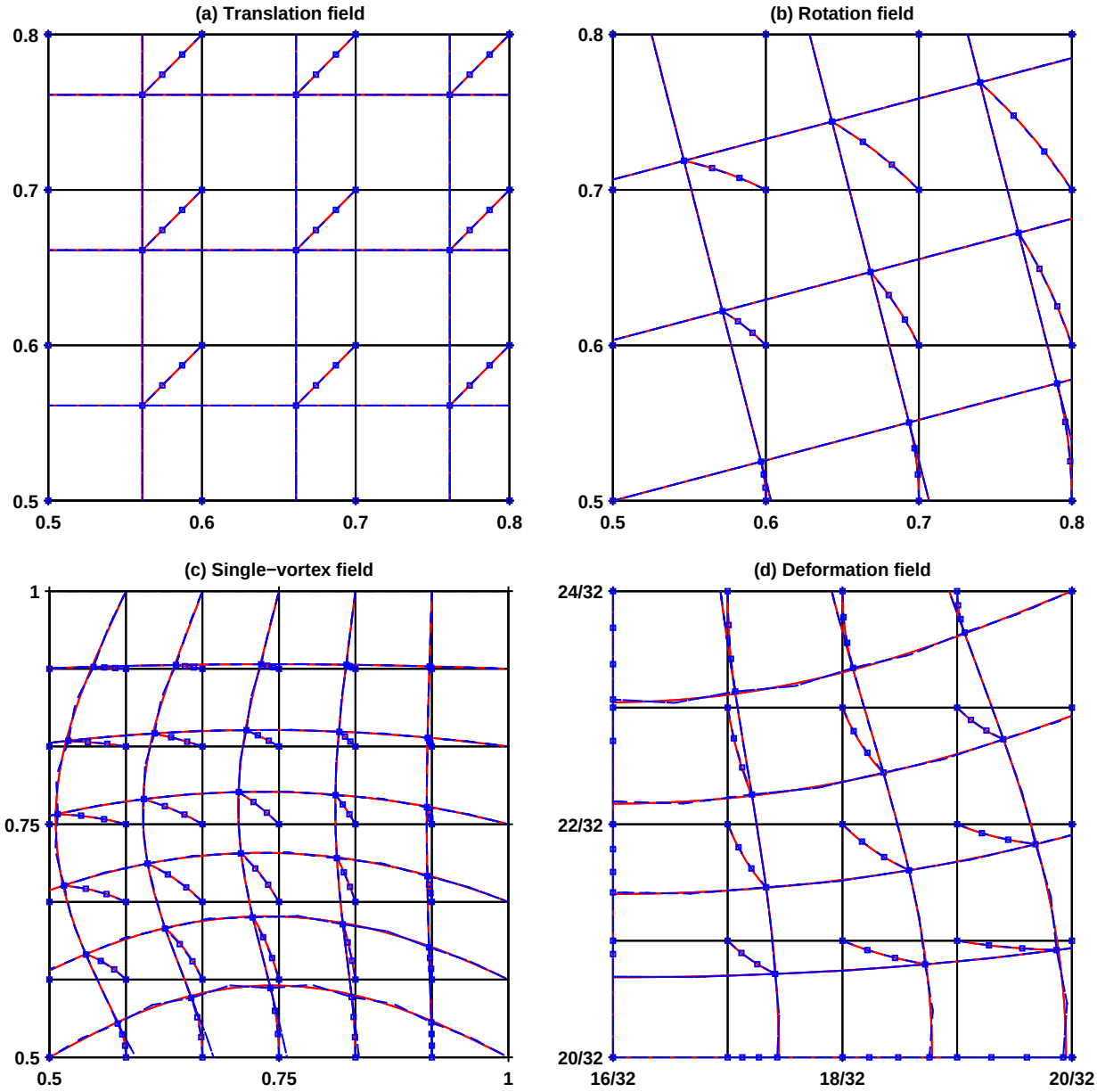


Figure 5.12: The grid's pre-image and vertex's streaklines on representative unit-cells of the velocity field: (a) in simple translation, (b) in simple rotation, (c) in the single vortex in-a-box field, (d) in the 4-vortex deformation field. The solid red lines are the exact curves, while the dashed blue lines are calculated lines. Square markers are the vertices of polygonal chains representing the streaklines.

Grid size	Courant number	Number of time step	Relative volume change
100 × 100	0.5	1072	5.10 %
	0.1	5113	1.25 %
150 × 150	0.5	1646	3.41 %
	0.1	8162	0.80 %

Table 5.2: Relative volume conservation error for the directional-split advection scheme.

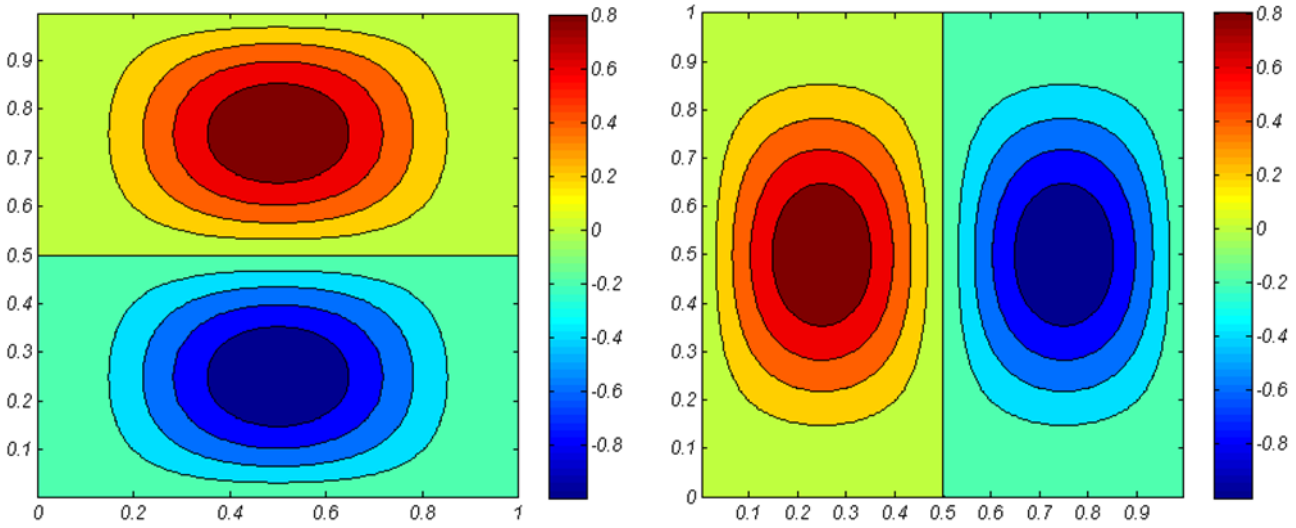


Figure 5.13: Non-uniform distribution of the velocity field, u component (left) and v component (right). The distribution is constant, but its amplitude varies in time.

The time-reversed single vortex flow is first tested for the direction-split advection scheme, on two uniform grids with the grid spacings 1/100 and 1/150, for the Courant numbers 0.5 and 0.1. The Figures 5.14–5.17 show snapshots of the material positions at different times. The red circles in the last pictures represent the theoretical final shape of the material, i.e. the initial disk. Finally, the Table 5.2 gives the accumulated volume conservation errors between the initial and final steps of the simulation. The volume is not conserved, and large geometrical errors are visible. At the end of the simulation, the center of mass of the deformed disk has moved to the left. This error is due to the explicit Euler scheme for the time integration. Thus, the geometric errors decreased with time-step refinement.

For the CCU scheme, the time-reversed single vortex test is performed on four uniform grids, with the grid spacings 1/32, 1/64, 1/128 and 1/256, see Figures 5.18 and 5.19 for snapshots of the simulations. Geometrical errors and convergence rates are compared to other unsplit advection schemes in Table 5.3. The geometrical error decreases with mesh refinements at a rate between 2 and 3. Overall, the CCU algorithm compares favorably to the other 2D unsplit advection schemes using the PLIC interface representation. In most of the cases, the CCU scheme decreases the geometrical errors by 15–30% as compared to the best of the other schemes, except for the coarsest mesh with the grid spacing 1/32. This could be due to the difference in the interface reconstruction algorithm. The benchmark tests of the other advection schemes use the LVIRA interface reconstruction algorithm, which is more accurate than the ELVIRA algorithm that we use, although both methods return similar results in most cases. In Figures 5.18 and 5.19, the theoretical position of the interface at the end of the simulation is represented by the red dashed line. We can see that the center of mass of the deformed disk is centered at the correct position at the end of the simulation, because of the Runge–Kutta time integration with quadratic interpolation in time. The Figures 5.18 and 5.19 show that the geometrical errors arise at the head and the tail of the deformed material. These regions of high geometrical errors are located where the interface has a high curvature, or forms thin filament below the thickness of 2 grid cells. In these regions, the interface reconstruction algorithm fails in capturing the interface accurately. The observed filament break-up is a numerical artifact, which should not be confused with actual fluid thread break-up due to minimization of the surface tension energy. This is not only a drawback of the LVIRA and the ELVIRA algorithm, but the PLIC representation in general. Therefore, the simulation results in the cases $T = 8$, for the coarse grids 32×32 and 64×64, are not very relevant to assess the accuracy of the CCU scheme versus other advection schemes, as the large geometrical errors come from the PLIC representation rather than the advection.

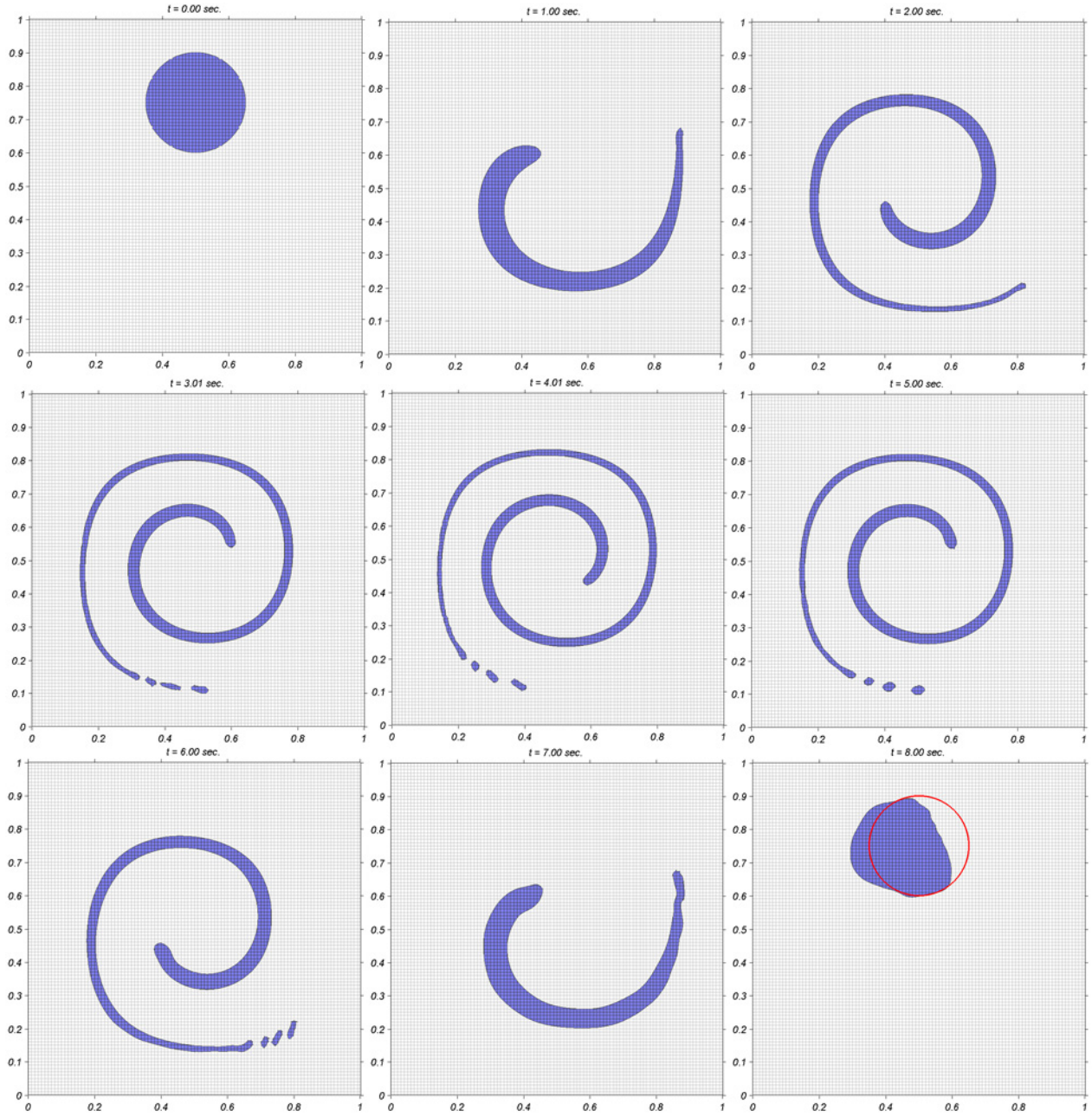


Figure 5.14: Simulation results for the directional-split advection scheme on the coarse grid (100x100), with the maximum Courant number 0.5. The red line shows the exact interface position at final time.

Small geometrical errors persist mainly because of errors in the interface representation. Poor interface representations (seen by substantial discontinuities of the interface) can generate advection errors. Figure 5.20 shows an example where advection errors are produced because of static interface reconstruction errors, in spite of an exact pre-images calculation. The use of more advanced interface representation techniques would be necessary in order to overcome this type of error. Quadratic interface reconstruction has been proposed by several authors (Scardovelli & Zaleski 2003; Renardy & Renardy 2002; Diwakar, Das & Sundararajan 2009; Vignesh & Bakshi 2013) to improve the representation of regions with high curvature. However, intersections of quadratic surfaces are more difficult to compute.

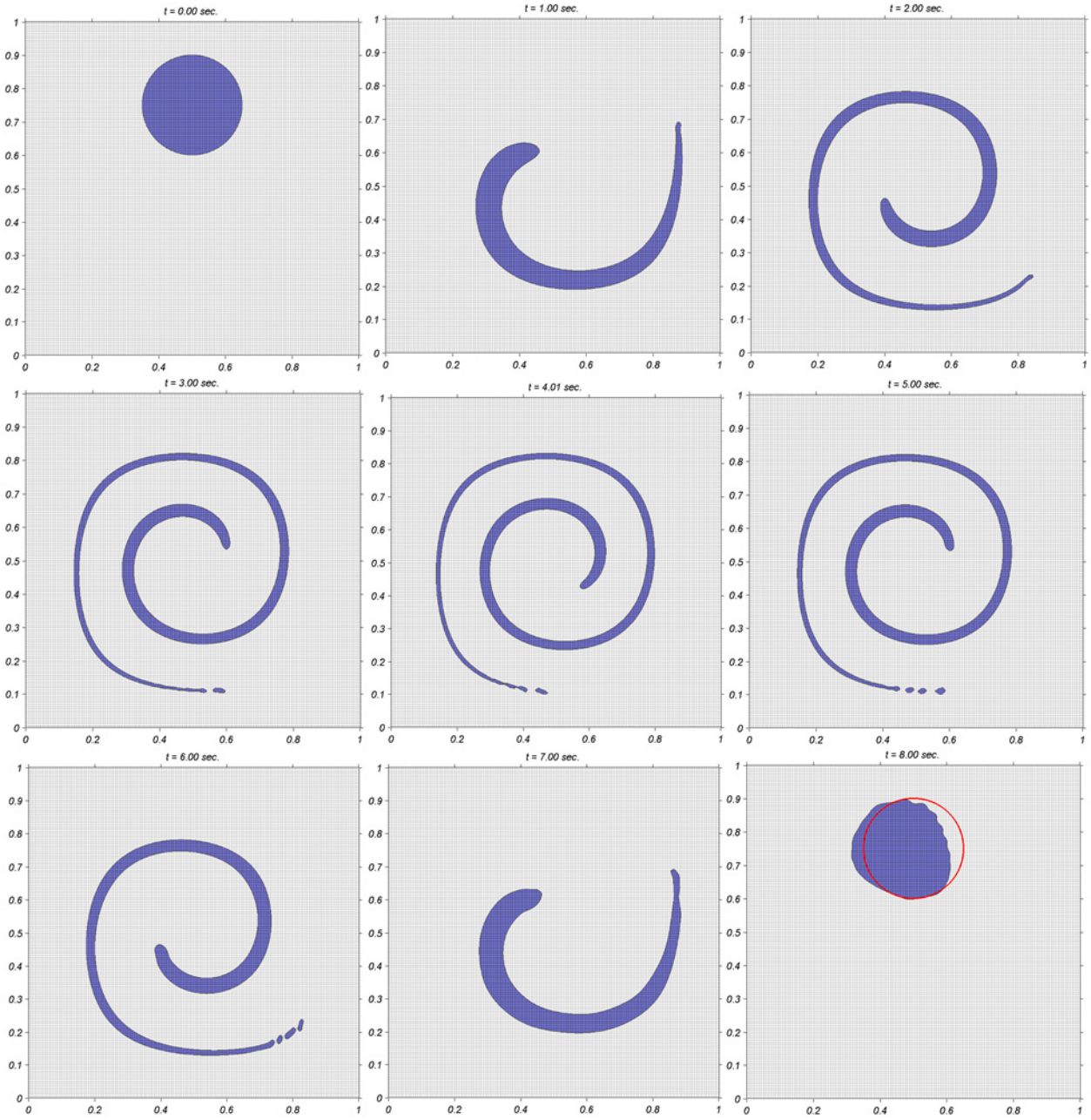


Figure 5.15: Simulation results for the directional-split advection scheme on the fine grid (150x150), with the maximum Courant number 0.5. The red line shows the exact interface position at final time.

Another technique, the improved PLIC-VOF method of López et al. (2005), successfully achieves the accurate representation of thin filaments with two non-secant line segments whose positions are tracked by markers. Both techniques are improvements of the interface representation and produce better results than the PLIC in the single vortex benchmark test. Other remaining issues of the PLIC-VOF method are (i) the improper representation of sharp corners, (ii) the interface reconstruction of multi-materials in cells containing tri-points, and (iii) the accurate tracking of sub-grid particles of fluid, such as small droplets or air bubbles. Some of these issues are handled better with other types of interface tracking methods, notably the hybrid markers-VOF method of Aulisa, Manservisi and Scardovelli (2003; 2004) and the PAM method of Zhang and Liu (2008).

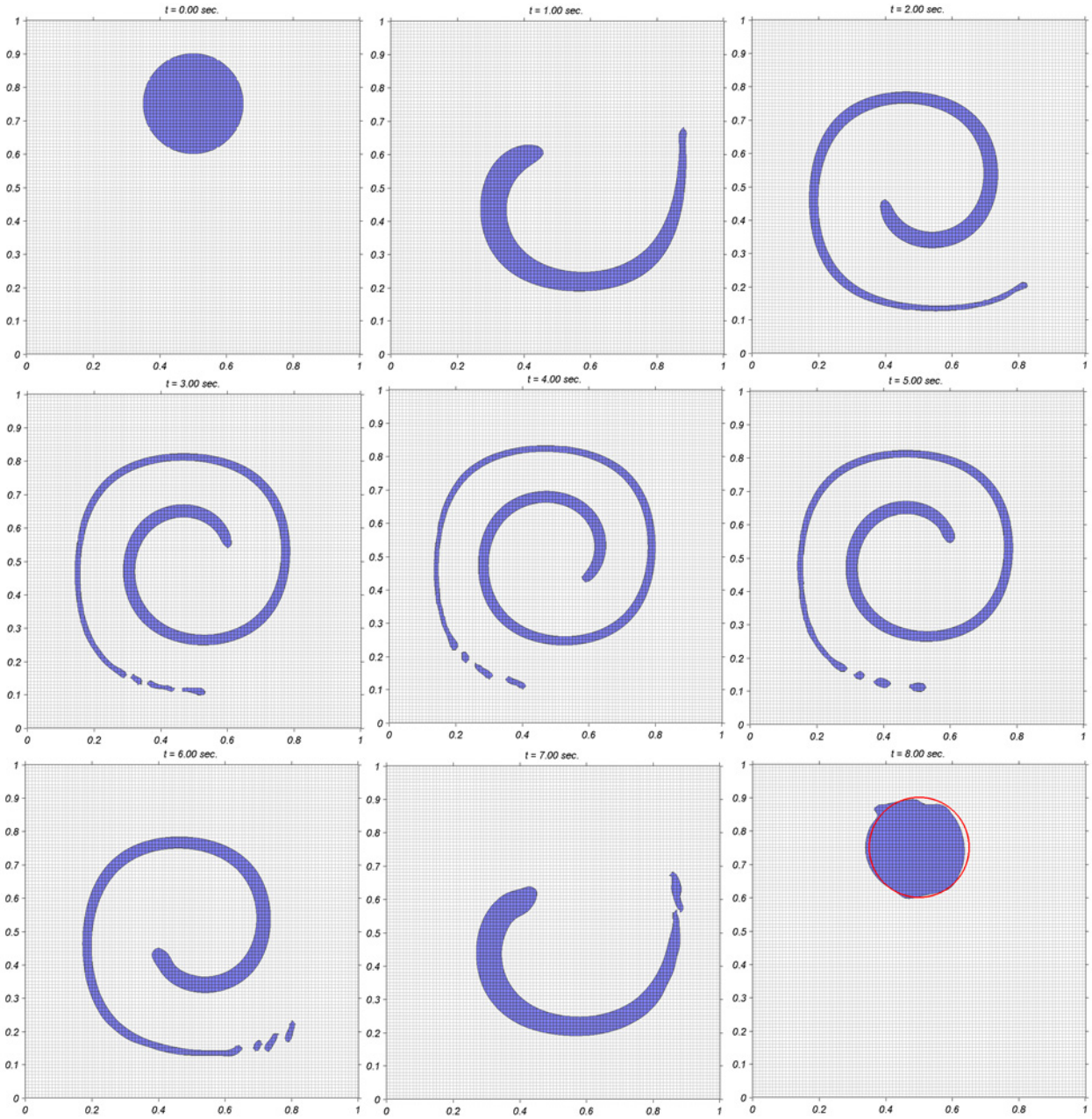


Figure 5.16: Simulation results for the directional-split advection scheme on the coarse grid (100x100), with the maximum Courant number 0.1. The red line shows the exact interface position at final time.

Both methods perform better than the PLIC-VOF when representing the interface, due to their use of Lagrangian geometrical objects such as markers or polygons. These two methods also have the ability to represent geometrical shapes below the grid resolution (bubbles or droplets). However, these improvements come at the price of tedious calculations. For instance, the hybrid marker-VOF method requires dynamical redistribution of the markers under the constraint of volume conservation, in order to keep a uniform marker density along the interface. The PAM method uses readjustment procedures, to shrink and expand the volume of polygons, and to reduce their number of vertices. Improvements of these readjustment procedures have been presented by Zhang and Fogelson (2014). In addition, both the original and the improved PAM methods require special algorithms to

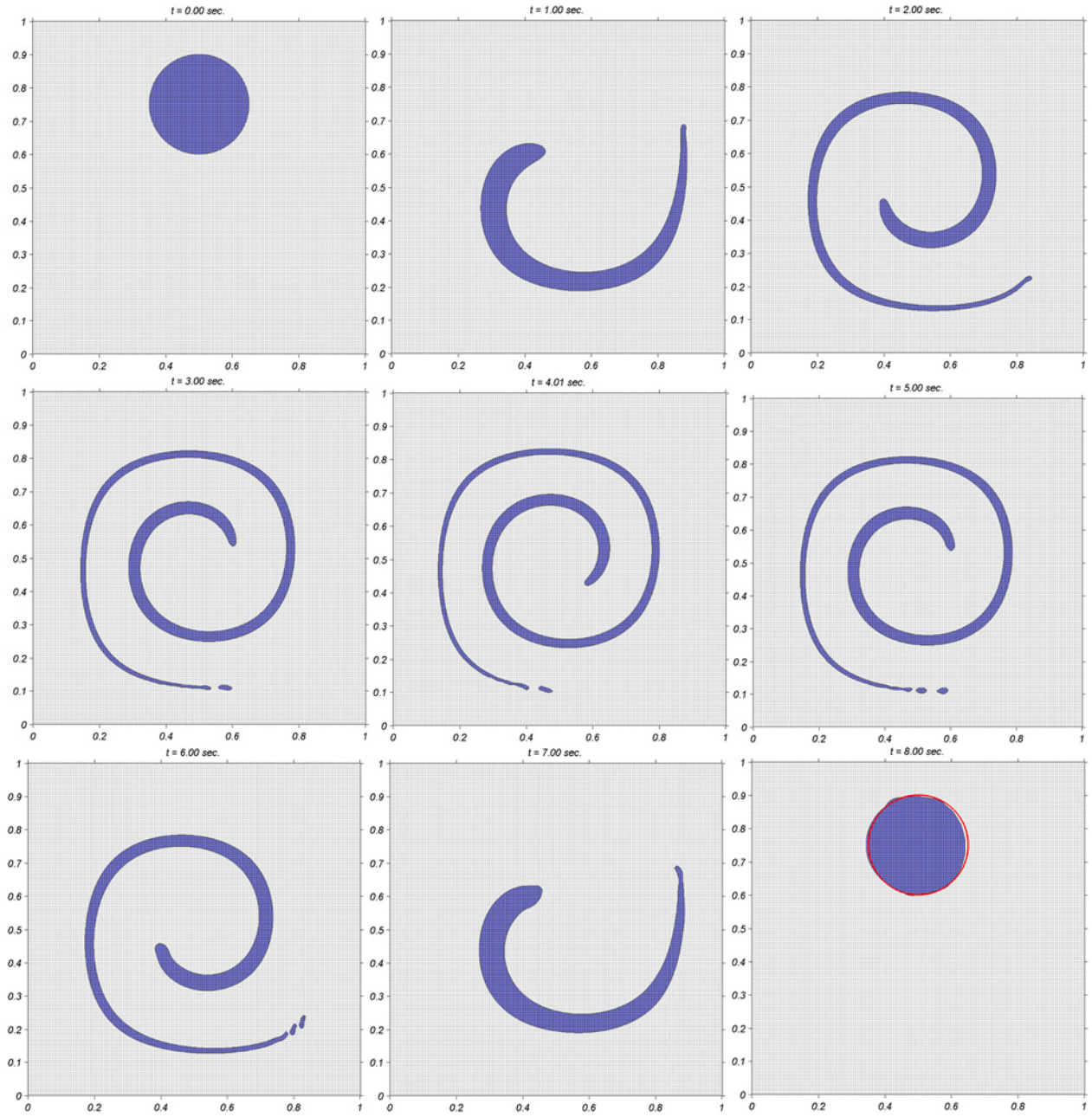


Figure 5.17: Simulation results for the directional-split advection scheme on the fine grid (150x150), with the maximum Courant number 0.1. The red line shows the exact interface position at final time.

successfully merge interfaces. On the contrary, Eulerian *surface capturing* techniques are relatively easier to implement and they handle interface merging automatically. Finally, the mentioned drawbacks of the PLIC-VOF method are rather due to the PLIC representation of the interface than the VOF approach itself. In the authors' opinion, the VOF method could reach equivalent accuracy as the Lagrangian methods, if more precise techniques for the interface representation were developed. In particular, we suggest the use of piecewise continuous bilinear interfaces, i.e. two-segment polygonal chains, to improve the representation of the sharp corners and the high curvatures.

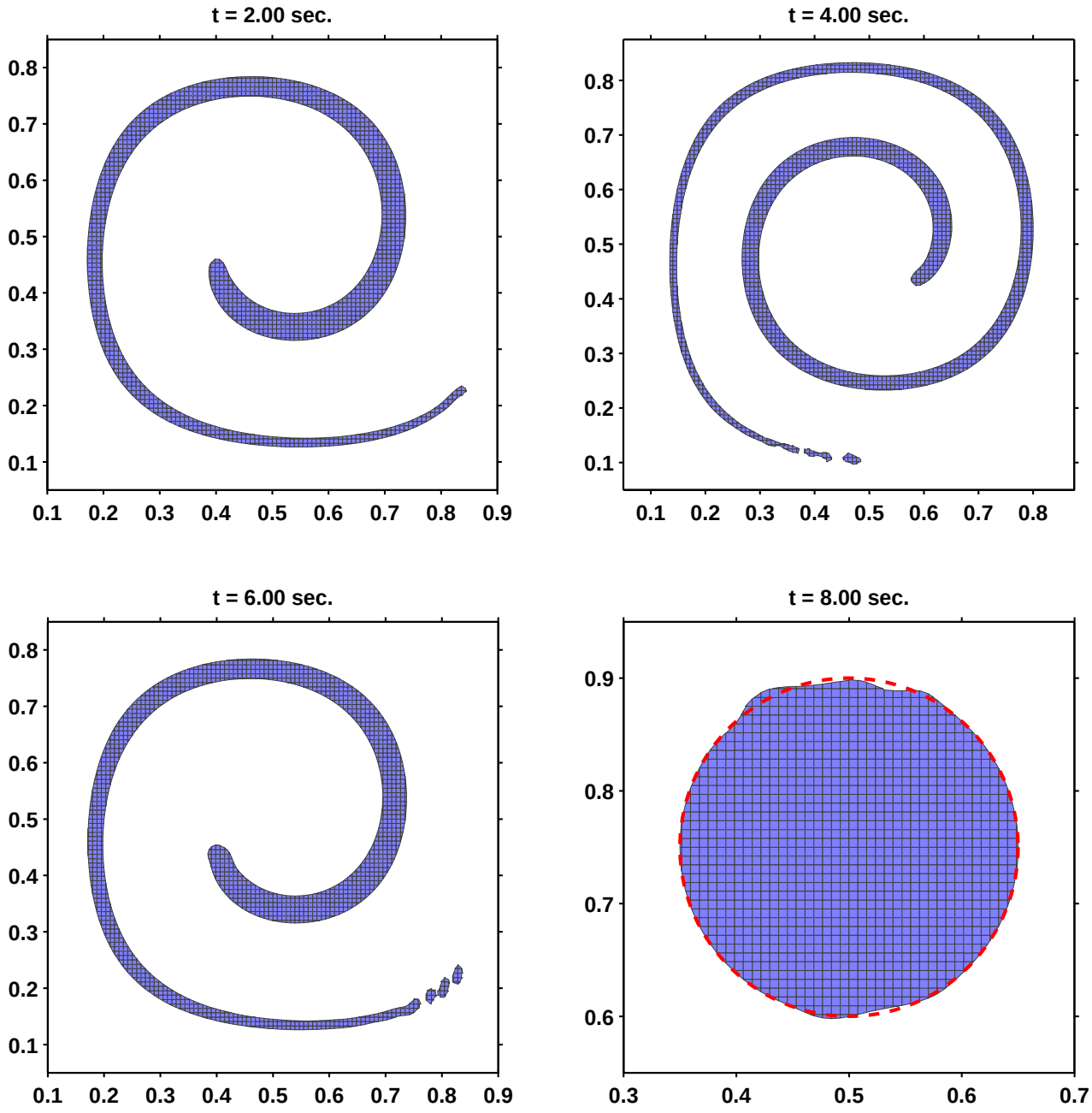


Figure 5.18: Reconstructed interfaces in the single vortex deformation test for the CCU advection scheme with $T = 8$ and $C_{\max} = 1$, for the 128×128 grid. Snapshots at $t = T/4$ (top-left), $t = T/2$ (top-right), $t = 3T/4$ (bottom-left) and $t = T$ (bottom-right). The red dashed line shows the exact interface position at final time. (Mind the different scales.)

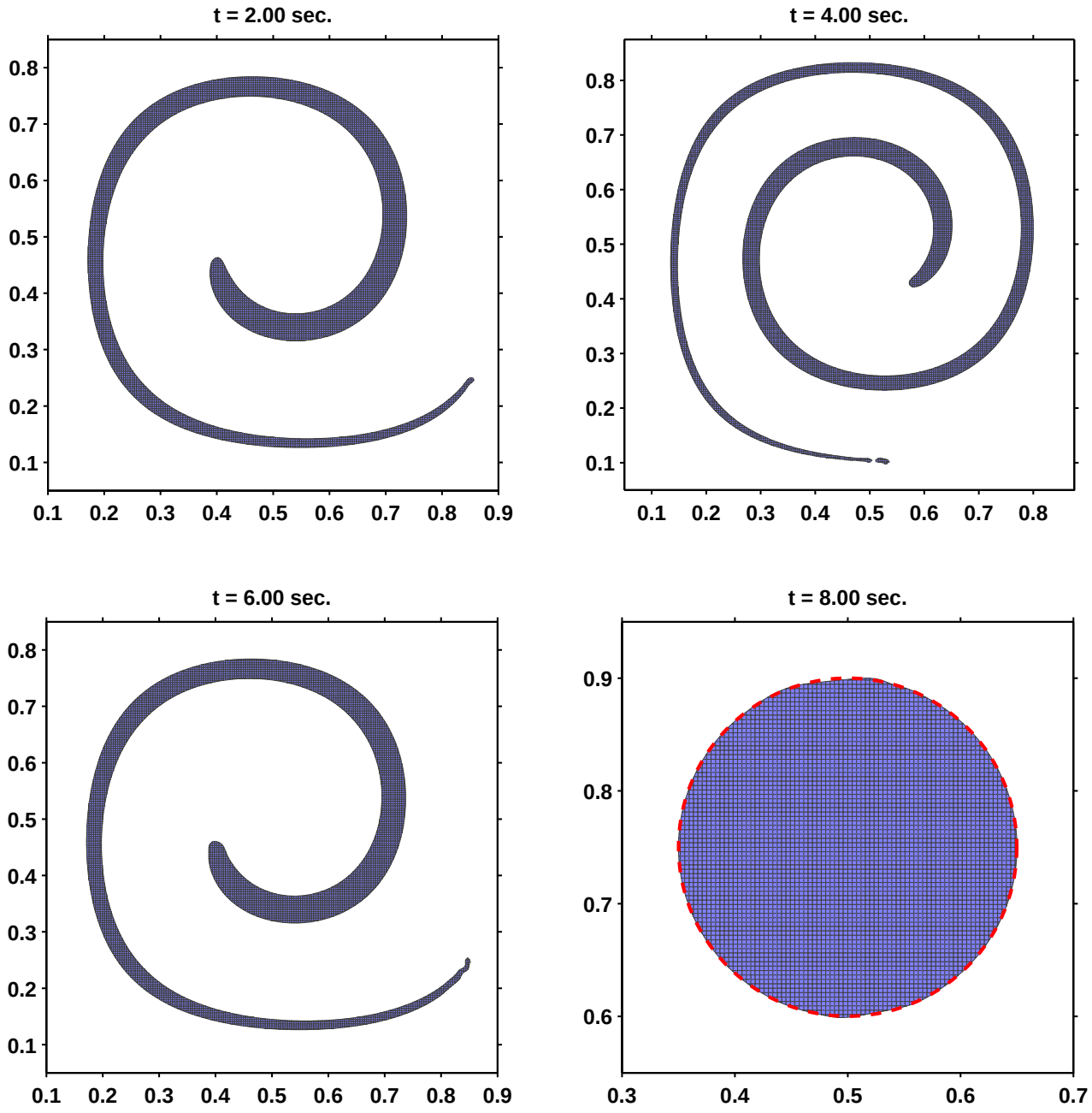


Figure 5.19: Reconstructed interfaces in the single vortex deformation test for the CCU advection scheme with $T = 8$ and $C_{\max} = 1$, for the 256×256 grid. Snapshots at $t = T/4$ (top-left), $t = T/2$ (top-right), $t = 3T/4$ (bottom-left) and $t = T$ (bottom-right). The red dashed line shows the exact interface position at final time. (Mind the different scales.)

	$E_g(32)$	$E_g(64)$	$O(32)$	$E_g(128)$	$O(64)$	$E_g(256)$	$O(128)$
Rider & Kothe (1998)	4.78 e-2	6.96 e-3	2.78	1.44 e-3	2.27	-	-
Harvie & Fletcher (2000)	3.72 e-2	6.79 e-3	2.45	1.18 e-3	2.52	-	-
López et al. (2004)	3.77 e-2	6.58 e-3	2.52	1.07 e-3	2.62	2.35 e-4	2.19
Mencinger & Zun (2011)	5.42 e-2	7.85 e-3	2.79	1.05 e-3	2.90	-	-
Cervones et al. (2009)	-	-	-	1.17 e-3	-	-	-
Owkes & Desjardins (2014)	-	7.58 e-3	-	1.88 e-3	2.01	4.04 e-4	2.22
CCU (proposed scheme)	3.81 e-2	4.58 e-3	3.06	1.00 e-3	2.20	1.78 e-4	2.59

Table 5.3: Geometrical errors and convergence rates in the single vortex deformation test cases for $T = 8$; comparison of the CCU scheme with other multi-dimensional advection schemes.

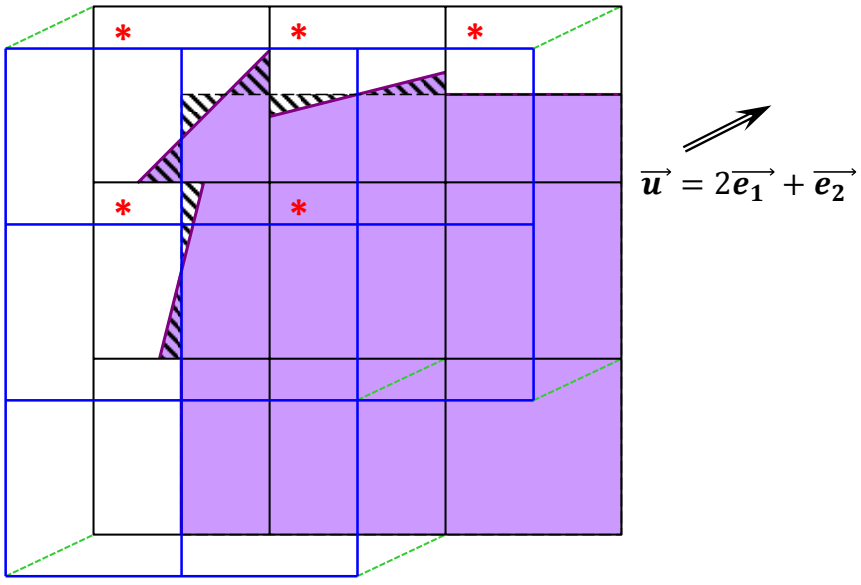


Figure 5.20: Advection errors in simple translation of a rectangular corner. The reconstructed shape of the material is colored in purple; hatched areas represent static reconstruction errors. The grid cells marked with a red asterisk are subjected to advection errors, in spite of the exact pre-image calculation in blue. Note that only the cells near the interface have their liquid volume fraction updated.

5.4. Rule of mixture

The conservation of momentum inside the interfacial cells is solved for the mixture of the two fluids as a whole. Therefore, it is necessary to evaluate apparent material properties of the mixture from the bulk material properties. Up to date, the majority of authors homogenize the material properties of the interfacial cells with a rule of mixture based on the arithmetic mean:

$$\tilde{\rho} = f\rho_1 + (1-f)\rho_2, \tag{5.36}$$

where ρ_1 and ρ_2 are the bulk densities of the two immiscible phases, and $\tilde{\rho}$ is the equivalent density of the mixture. This simple homogenization is correct for the density, because both mass and volume are scalar extensive physical quantities (i.e. additive). However, the viscosity is defined as the ratio of tensorial quantities (stress over strain rate). The issue of selecting a correct rule of mixture for the equivalent viscosity $\tilde{\eta} = \tilde{\eta}(\eta_1, \eta_2, f)$, where η_1 and η_2 are the bulk viscosities of the two liquid phases, has been raised by several authors. Tryggvason et al. (1998) emphasized that both the arithmetic and harmonic means could be appropriate, depending on whether the velocity gradient (i.e. the strain rate) or the viscous flux (i.e. the viscous stress) are continuous across the interface. The first choice enhances the role of the more viscous phase in the dynamics of the mixture, while the latter promote the influence of the less viscous material (Kothe 1998). A compromise consists in averaging the apparent viscosities related to the tensile viscous stresses τ_{xx} and τ_{yy} with an arithmetic mean, and the apparent viscosity related to the shear stress τ_{xy} with a harmonic mean (Rudman 1998). Coward et al. (1997) used those averages for the numerical simulation of interfacial instabilities driven by the shear-rate discontinuity, in stratified Couette flows. Their results clearly show that the choice of the viscosity average has an impact on the solution. Nevertheless, they pointed out that the averages they use are physically correct only when the interface is close to be aligned with the grid (which was the case in their simulations), and they acknowledged the need for a more general method applicable to interfaces with arbitrary orientations. The use of the correct viscosity averages is crucial for the accurate simulations of multiphase flows.

A new rule of mixture has been derived based on the decompositions of the viscous stresses into parallel and perpendicular components to the interface. The basic idea is that the apparent viscosity should not only depend on the liquid volume fraction f, but also on the orientation of the interface with respect to the principal direction of the deformations. There is an analogy between the mixture of the two phases and a fiber-reinforced composite material, where the effective stiffness depends on the orientation of the fibers with the direction of loading.

We first consider two special configurations of the interface, depicted in Figure 5.21. In the first case, shear deformations are applied orthogonally to the interface; therefore, the two phases experience the same shear-rate $\dot{\gamma} = \partial u/\partial y + \partial v/\partial x$. There is continuity in the normal velocity gradient at the interface. Consequently, the two fluids behave in parallel, and the total stress response is the sum of the shear stress built-up in each of the phases. Therefore, the equivalent parallel viscosity η^{P} of the mixture is obtained with the *arithmetic mean*:

$$\eta^{\mathrm{P}} = f\eta_1 + (1-f)\eta_2. \tag{5.37}$$

Viscosities in parallel (arithmetic mean)

$$\eta^{\mathrm{P}} = f\eta_1 + (1-f)\eta_2$$

Viscosities in serial (harmonic mean)

$$\eta^{\mathrm{S}} = \frac{1}{\dfrac{f}{\eta_1} + \dfrac{1-f}{\eta_2}}$$

Figure 5.21: Parallel and serial viscosity of the mixture, obtained by the arithmetic and harmonic means.

In the second configuration, shear deformations are applied in parallel to the interface, thus the two phases experience different deformation rates. However, there is equilibrium in the shear stresses τ_{xy} on both sides of the interface, due to the continuity of the viscous flux across the interface. Therefore, the two fluids react in series, and the correct equivalent series viscosity η^{S} is obtained with the *harmonic mean*:

$$\eta^{\mathrm{S}} = \left(\frac{f}{\eta_1} + \frac{1-f}{\eta_2} \right)^{-1} . \tag{5.38}$$

In the general case with an arbitrary orientation of the interface, the parallel and series viscosities η^{P} and η^{S} are the two extrema (upper and lower bounds, respectively) of the possible apparent viscosity, varying with the principal direction of deformations. Hence, in our approach, different apparent viscosities $\tilde{\eta}_{ij}$ are evaluated for each component of the viscous stress tensor:

$$\tau_{ij} = 2\tilde{\eta}_{ij} D_{ij} . \tag{5.39}$$

Each component of the strain-rate tensor D_{ij} can be decomposed into the sum of its normal and tangential projections on the interface, $D_{ij}^{\perp}$ and $D_{ij}^{\parallel}$ respectively, such that:

$$D_{ij}^{\perp} = D_{ij} \cos\chi , \tag{5.40}$$

$$D_{ij}^{\parallel} = D_{ij} \sin\chi , \tag{5.41}$$

and

$$D_{ij} = \left[\left(D_{ij}^{\perp} \right)^2 + \left(D_{ij}^{\parallel} \right)^2 \right]^{1/2} , \tag{5.42}$$

where χ is the angle between the normal vector $\hat{\boldsymbol{n}}$ of the interface and the direction of the component D_{ij} of the strain rate tensor, see Figure 5.22. The components of the stress tensor are decomposed similarly:

$$\tau_{ij} = \left[\left(\tau_{ij}^{\perp} \right)^2 + \left(\tau_{ij}^{\parallel} \right)^2 \right]^{1/2} . \tag{5.43}$$

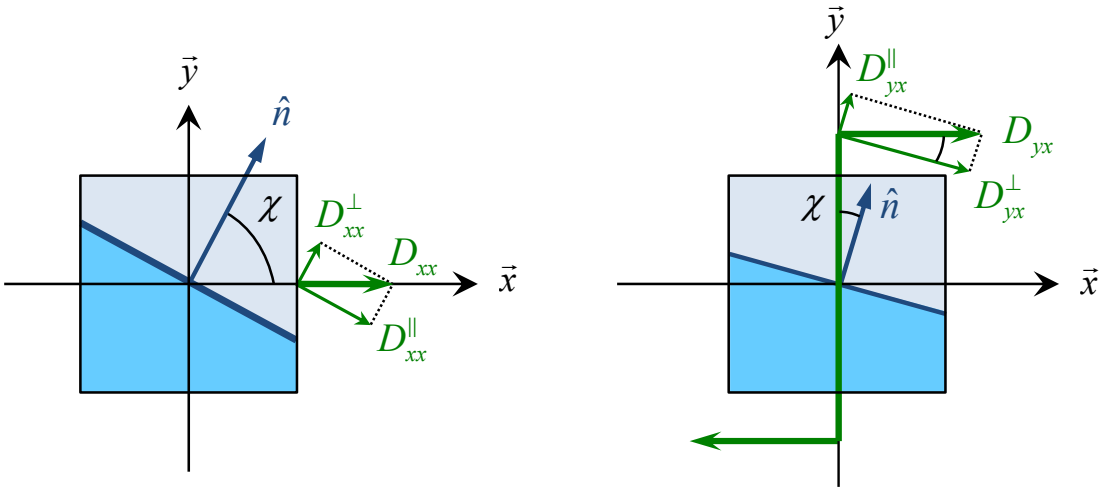


Figure 5.22: Orthogonal projections of the direction of the rate-of-deformations on the normal and tangential directions to the reconstructed interface.

Because of linear superposition of the stresses and the strain rates, we have:

$$\tau_{ij}^{\perp} = 2\eta_{ij}^{\mathrm{S}} D_{ij}^{\perp}, \tag{5.44}$$

$$\tau_{ij}^{\parallel} = 2\eta_{ij}^{\mathrm{P}} D_{ij}^{\parallel}. \tag{5.45}$$

Thus, recalling equations (5.39)–(5.45), we deduce the following formula for the component $\tilde{\eta}_{ij}$ of the apparent viscosities of mixture:

$$\tilde{\eta}_{ij} = \left[\left(\eta_{ij}^{\mathrm{S}} \cos\chi \right)^2 + \left(\eta_{ij}^{\mathrm{P}} \sin\chi \right)^2 \right]^{1/2}. \tag{5.46}$$

Therefore, the equivalent viscosity of the mixture is the *quadratic mean* of the arithmetic and harmonic means of the two bulk viscosities, weighed by the ratio of the parallel to in-series deformations with respect to the interface.

In the case of free-surface simulations (where one of the fluid phases represents the air), the viscosity of the air is set to a thousandth of the viscosity of the other phase, in order to improve the numerical stability. The artificially high viscosity of the air affect the flow of the air, but it does not impact the flow of the most viscous phase.

Finally, as the PLIC–VOF method is not able to resolve the flow details below the grid resolution, the free-surface flow of a viscous fluid inside a channel is affected by a numerical artefact near the wall. In the simulation, the viscous fluid that rolls on the wall does not remove all the air inside the interfacial cells at the wall. As it can be seen in Figure 5.23, a thin layer of air gets trapped between the wall and the viscous fluid. Consequently, the quadratic mean (5.46) is ineffective for the interfacial cells at the wall. Indeed, the quadratic mean (5.46) would predict the apparent viscosity as the harmonic mean η^{S} of the viscosities, which is very close to the viscosity of the air. This results in a wrong shear rate at the wall. Hence, the thin layer of air that is artificial trapped would act on the wall as a lubricant, and the viscous material would flow as a plug flow. In order to remedy this numerical artefact, the apparent viscosities in the interfacial cells at the wall are taken as the viscosity of the dominant phase. Thus, the layer of air does not affect the shear-rate at the wall. This correction of the viscosity averages at the wall has been used in the computation of the free surface flow in a rectangular channel, represented in Figure 5.23. The numerical experiments presented in the next chapter utilize the new quadratic mean of the bulk viscosities as the rule of mixture, together with the viscosity wall correction.

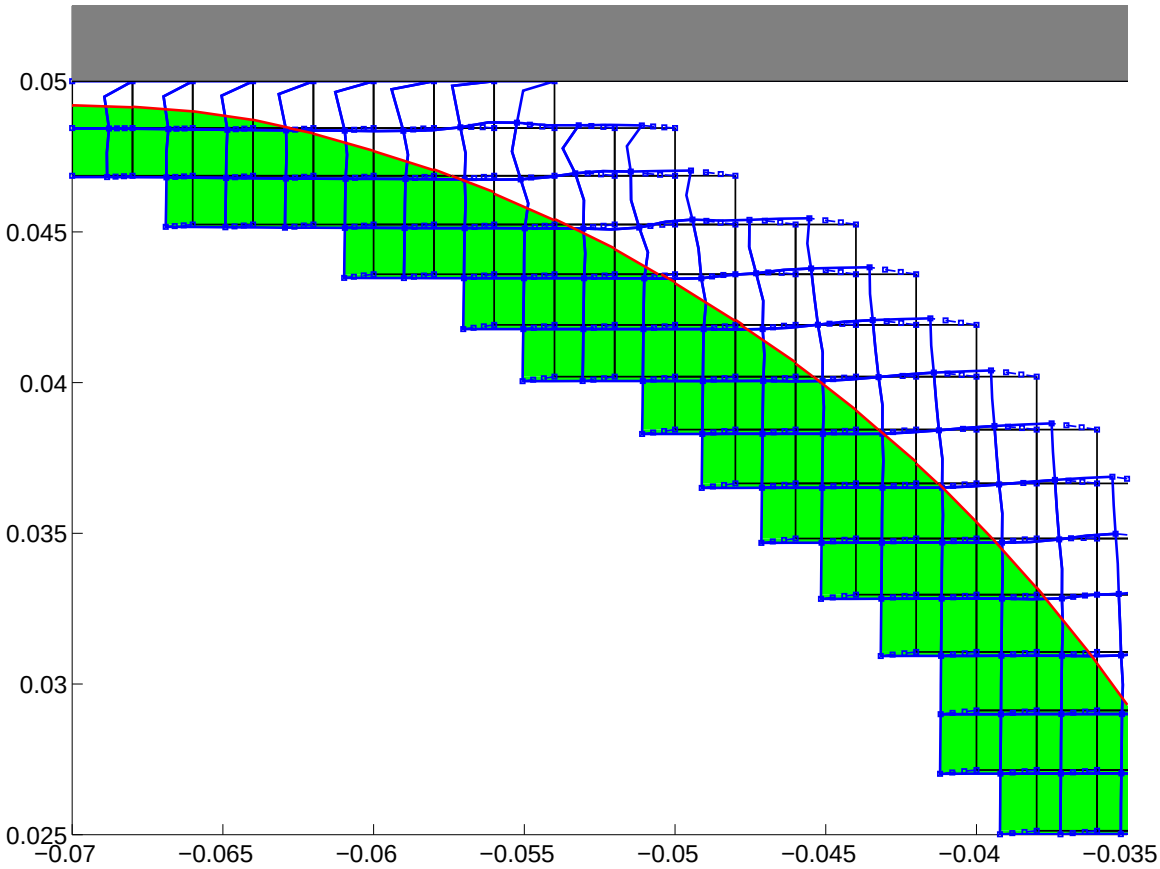


Figure 5.23: Free-surface flow of a viscous fluid inside a rectangular channel (with correction of the viscosity averages at the wall). The grid lines of the interfacial cells are in black; the streaklines and the pre-images of the interfacial cells are in blue; the reconstructed interface is in red, the viscous fluid inside the cells near the interface is represented in green.

5.5. References

Aulisa, E, Manservisi, S & Scardovelli, R 2003, 'A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows', *Journal of Computational Physics*, vol. 188, pp. 611–639.

Aulisa, E, Manservisi, S, Scardovelli, R & Zaleski, S 2003, 'A geometrical area-preserving volume-of-fluid advection method', *Journal of Computational Physics*, vol. 192, pp. 355–364.

Aulisa, E, Manservisi, S & Scardovelli, R 2004, 'A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking', *Journal of Computational Physics*, vol. 197, pp. 555–584.

Aulisa, E, Manservisi, A, Scardovelli, R & Zaleski, S 2007, 'Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry', *Journal of Computational Physics*, vol. 225, pp. 2301–2319.

Cervone, A, Manservisi, A, Scardovelli, R & Zaleski, S 2009, 'A geometrical predictor–corrector advection scheme and its application to the volume fraction function', *Journal of Computational Physics*, vol. 228, pp. 406–419.

Coward, AV, Renardy, YY, Renardy, M & Richards, JR 1997, 'Temporal evolution of periodic disturbances in two-layer Couette flow', *Journal of Computational Physics*, vol. 132, pp. 346–361.

Diwakar, SV, Das, SK & Sundararajan, T 2009, 'A quadratic spline based interface (QUASI) reconstruction algorithm for accurate tracking of two-phase flows', *Journal of Computational Physics*, vol. 228, pp. 9107–9130.

Dyadechko, V & Shashkov, M 2005, *Moment-of-fluid interface reconstruction*, Los Alamos report LA-UR-07-1537, Los Alamos National Laboratory.

Ginzburg, I & Wittum, G 2001, 'Two-phase flows on interface refined grids modeled with VOF, staggered finite volumes, and spline interpolants', *Journal of Computational Physics*, vol. 166, pp. 302–335.

Harvie, DJE & Fletcher, DF 2000, 'A new volume of fluid advection algorithm: the stream scheme', *Journal of Computational Physics*, vol. 162, pp. 1–32.

Hirt, CW & Nichols, BD 1981, 'Volume of fluid (VOF) method for the dynamics of free boundaries', *Journal of Computational Physics*, vol. 39, pp. 201–225.

Kothe, DB 1998, *Perspective on Eulerian finite volume methods for incompressible interfacial flows*, Courses and lectures, International Centre for Mechanical Sciences.

Pilliod Jr, JE & Puckett, EG 2004, 'Second-order accurate volume-of-fluid algorithms for tracking material interfaces', *Journal of Computational Physics*, vol. 199, pp. 465–502.

Puckett, EG, Almgren, AS, Bell, JB, Marcus, DL & Rider, WJ 1997, 'A high-order projection method for tracking fluid interfaces in variable density incompressible flows', *Journal of Computational Physics*, vol. 130, pp. 269–282.

López, J, Hernández, Gómez, P & Faura, F 2004, 'A volume of fluid method based on multidimensional advection and spline interface reconstruction', *Journal of Computational Physics*, vol. 195, pp. 718–742.

López, J, Hernández, J, Gómez, P & Faura, F 2005, 'An improved PLIC-VOF method for tracking thin fluid structures in incompressible two-phase flows', *Journal of Computational Physics*, vol. 208, pp. 51–74.

Mencinger, J & Zun, I 2011, 'A PLIC–VOF method suited for adaptive moving grids', *Journal of Computational Physics*, vol. 230, pp. 644–663.

Osher, S & Fedkiw, RP 2001, 'Level set methods: an overview and some recent results', *Journal of Computational Physics*, vol. 169, pp. 463–502.

Owkes, M & Desjardins, O 2014, 'A computational framework for conservative, three-dimensional, unsplit, geometric transport with application to the volume-of-fluid (VOF) method', *Journal of Computational Physics*, vol. 270, pp. 587–612.

Renardy, Y & Renardy, M 2002, 'PROST: a parabolic reconstruction of surface tension for the volume-of-fluid method', *Journal of Computational Physics*, vol. 183, pp. 400–421.

Rudman, M 1998, 'A volume-tracking method for incompressible multifluid flows with large density variations', *International Journal for Numerical Methods in Fluids*, vol. 28, pp. 357–378.

Scardovelli, R & Zaleski, S 1999, 'Direct numerical simulation of free-surface and interfacial flow', *Annual Review of Fluid Mechanics*, vol. 31, pp. 567–603.

Scardovelli, R & Zaleski, S 2003, 'Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection', *International Journal for Numerical Methods in Fluids*, vol. 41, pp. 251–274.

Sethian, JA & Smereka, P 2003, 'Level set methods for fluid interfaces', *Annual Review of Fluid Mechanics*, vol. 35, pp. 341–372.

Tryggvason, G, Bunner, B, Ebrat, O & Tauber, W 1998, *Computations of multiphase flows by a finite difference/front tracking method. I. Multi-fluid flows*, Lecture series, Von Karman Institute for Fluid Dynamics.

Tryggvason, G, Bunner, B, Esmaeeli, A, Juric, D, Al-Rawahi, N, Tauber, W, Han, J, Nas, S & Jan, Y-J 2001, 'A front-tracking method for the computations of multiphase flow', *Journal of Computational Physics*, vol. 169, pp. 708–759.

Vignesh, TG & Bakshi, S 2013, 'Noniterative interface reconstruction algorithms for volume of fluid method', *International Journal for Numerical Methods in Fluids*, vol. 73, pp. 1–18.

Weymouth, GD & Yue, DK-P 2010, 'Conservative Volume-of-Fluid method for free-surface simulations on Cartesian-grids', *Journal of Computational Physics*, vol. 229, pp. 2853–2865.

Youngs, DL 1982, 'Time-dependent multi-material flow with large fluid distortion', in: *Numerical Methods for Fluid Dynamics*, pp. 273–285, Academic Press, New York.

Zhang, Q & Liu, PL-F 2008, 'A new interface tracking method: The polygonal area mapping method', *Journal of Computational Physics*, vol. 227 pp. 4063–4088.

Zhang, Q 2013, 'On donating regions: Lagrangian flux through a fixed curve', *SIAM Review*, vol. 55, pp. 443–461.

Zhang, YL, Yeo, KS, Khoo, BC & Wang, C 2001, '3D jet impact and toroidal bubbles', *Journal of Computational Physics*, vol. 166, pp. 336–360.

Zhang, Q 2013, 'Highly accurate Lagrangian flux calculation via algebraic quadratures on spline-approximated donating regions', *Computer Methods in Applied Mechanics and Engineering*, vol. 264, pp. 191–204.

Zhang, Q 2013, 'On a family of unsplit advection algorithms for volume-of-fluid methods', *SIAM Journal on Numerical Analysis*, vol. 51, pp. 2822–2850.

Zhang, Q & Fogelson, A 2014 'Fourth-order interface tracking in two dimensions via an improved polygonal area mapping method', SIAM Journal on Scientific Computing, vol. 36, pp. A2369–A2400.

Chapter 6

Numerical results

The numerical algorithms described in the previous chapters have been implemented in Matlab. The present chapter reports numerical results of viscoelastic flows in benchmark test cases, using the *streamfunction/log–conformation* method, described through chapters 2, 3 and 4. The single-phase viscoelastic flows of an Oldroyd–B fluid are solved in the lid-driven cavity, and the 4:1 planar contraction. Multi-phase simulations are performed to compute the free-surface flows of an Oldroyd–B fluid exiting a planar slit die. The multi-phase flow simulations use the PLIC–VOF method with the ELVIRA interface reconstruction algorithm and the *Cellwise Conservative Unsplit* (CCU) advection scheme, presented in chapter 5.

6.1. Single-phase viscoelastic flows in a lid-driven cavity

The viscoelastic flow in a lid-driven cavity has been used as a benchmark test case in the literature to evaluate the performance of viscoelastic flow solvers (Fattal & Kupferman 2004; Fattal & Kupferman 2005; Pan, Hao & Glowinski 2009; Hao & Pan 2007; Belblidia at al. 2007). The geometry consists in a square cavity with a unit length $L=1$. The flow is driven by the tangential motion of the upper wall, while the other boundaries are stationary, with the no-slip condition, see Figure 6.1. The tangential velocity profile of the lid is regularized (Fattal & Kupferman 2005):

$$u_t(x,t)=16x^2(1-x)^2\left(0.5+\tanh\left(8(t-0.5)\right)/2\right)U,\quad x\in[0,1],\quad t>0, \tag{6.1}$$

in order to remove the stress singularity at the corners (Renardy 2003). In this way, the stagnation points located at the corners are not subjected to any deformation rate, preventing the appearance of infinitely increasing stress gradients, which could compromise the smoothness of the solution (Renardy 2006). The regularized velocity profile can be interpreted physically as the result of the fluid slippage on the moving lid, as shown by molecular dynamics simulations (Koplik & Banavar 1995). In addition, the hyperbolic tangent function smoothes the acceleration of the lid, and removes the sudden deformation rate at the initial time. The lid velocity has reached its nominal values when $t>1$. The maximum lid velocity is set to $U=1$. In this setup, the Weissenberg number is defined as $Wi=\lambda U/L$.

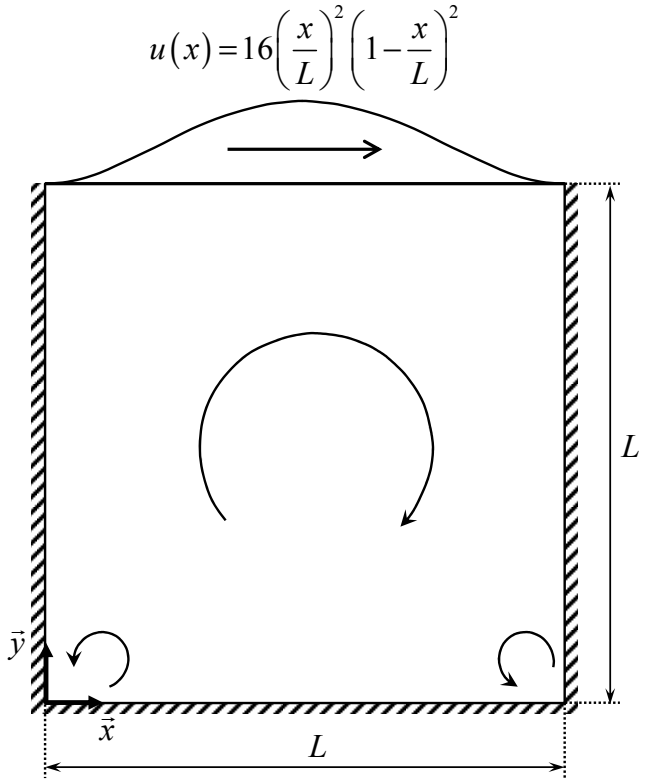


Figure 6.1: Geometry of the regularized lid-driven cavity.

In spite of the simple geometry, this problem is considered as particularly difficult to solve, especially at high Weissenberg numbers, since the flow exhibits complex features combining shear and extensional deformations. For $Wi = 0.25$, Figure 6.2(left) represents the flow classifier ϑ, which is defined as:

$$\vartheta \equiv \frac{1-R_S}{1+R_S}, \tag{6.2}$$

where

$$R_S = \frac{|\omega/2|}{\sqrt{\dot{\varepsilon}^2+\dot{\gamma}^2}} \tag{6.3}$$

is the persistence-of-straining parameter (Thompson & Mendes 2005), ω is the vorticity, $\dot{\varepsilon} \equiv \partial u/\partial x$ is the extensional strain rate, and $\dot{\gamma} \equiv \left(\partial u/\partial y + \partial v/\partial x\right)/2$ is the shear rate. The flow classifier has the following properties: $\vartheta = 1$ $\left(R_S = 0\right)$ in pure extensional flow, $\vartheta = 0$ $\left(R_S = 1\right)$ in pure shear flow, and $\vartheta = -1$ $\left(R_S \to \infty\right)$ in pure rigid-body rotation (ϑ and R_S are indefinite when the flow is at rest or in pure translation). The magnitude of the extensional strain rate is also depicted in Figure 6.2(right), for $Wi = 0.25$. Figure 6.2 shows that the material successively experiences shear and extensional deformations. Moreover, the extensional rate reaches large values near the moving lid. As the cavity is a closed system (without inlets and outlets), there is a critical Weissenberg number above which the recirculating material build-up extra-stresses faster than it relaxes. Then, the flow cannot reach a steady state solution, and it exhibits elastic instabilities. In this scenario, the flow develops steep stress boundary layers that require a very fine resolution, otherwise the under-resolution produces numerical instabilities (Fattal & Kupferman 2005).

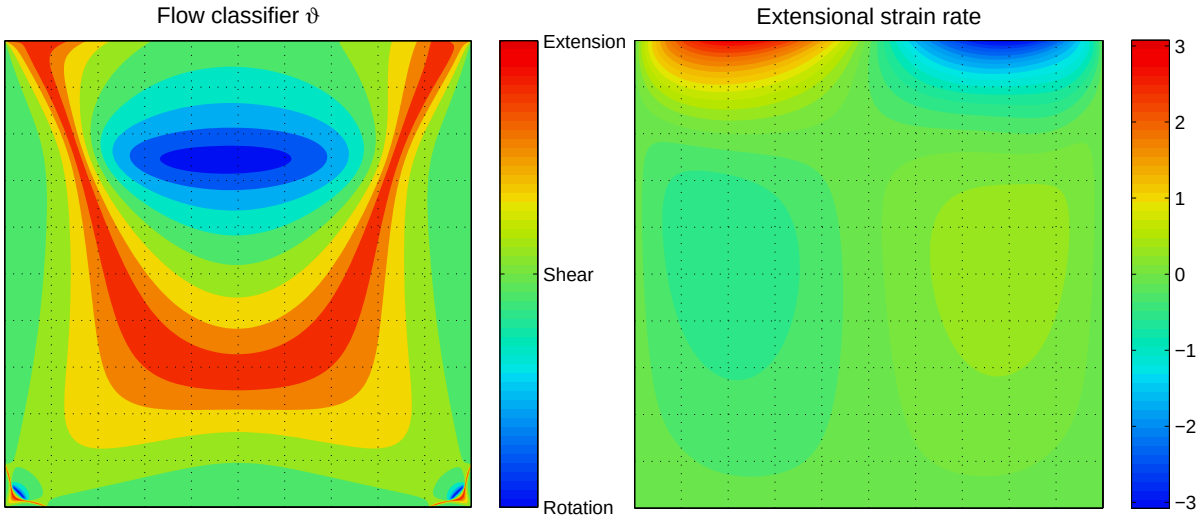


Figure 6.2: Flow classifier ϑ and extensional strain rate $\dot{\varepsilon}$ of the steady-state solution, for $Wi = 0.25$.

Moreover, in the case of an Oldroyd–B fluid, which lacks finite extensibility, an unbounded stress growth occurs under steady extensional deformations, when $\dot{\varepsilon} > 1/(2\lambda)$, as a manifestation of the dissipative constitutive instability. Thus, from a numerical point of view, the simulation of the viscoelastic lid-driven cavity flow at high Weissenberg numbers is similarly challenging as its Newtonian counterpart at high Reynolds numbers.

The regularized lid-driven cavity flow of the Oldroyd–B fluid is simulated for various Weissenberg numbers (up to $Wi = 10$) by varying the relaxation times. A constant solvent viscosity ratio $\beta = 1/2$ is used. The Reynolds number is set to the negligible value $Re = \rho UL/\eta_0 = 5 \cdot 10^{-4}$, such that the flow is virtually considered as a creeping flow.

6.1.1. Orders of convergence

In this subsection, where the spatial and temporal order of convergences are assessed, the residual tolerance of the iterations algorithm is set to the value $\varepsilon_{\text{tol}} = 10^{-12}$. The spatial convergence rate is estimated from the relative errors with respect to successive mesh refinements, since there is no analytical solution to evaluate the absolute errors. The simulation is carried out on a Cartesian mesh with the grid spacings $h = 1/40$, $1/80$, $1/160$ and $1/320$, and with a fixed time step size $\Delta t = 10^{-2}$. The relative error is locally monitored on a sample of 81 points evenly distributed inside the cavity (whose locations correspond to the intersections of the grid lines plotted in Figure 6.2). For a grid size h, the relative error $e_r(h, x_i, y_j)$ of the numerical solution $\chi_h(x_i, y_j)$ at the point with coordinates $(x_i, y_j) = (iL/10, jL/10)$ is:

$$e_r(h, x_i, y_j) = \left| \chi_{2h}(x_i, y_j) - \chi_h(x_i, y_j) \right|. \tag{6.4}$$

A norm for the global relative error is defined as the root-mean-square of all the local relative errors:

$$E_r(h) = \left[\frac{1}{81} \sum_{i,j=1}^{9} e_r(h, x_i, y_j)^2 \right]^{1/2}. \tag{6.5}$$

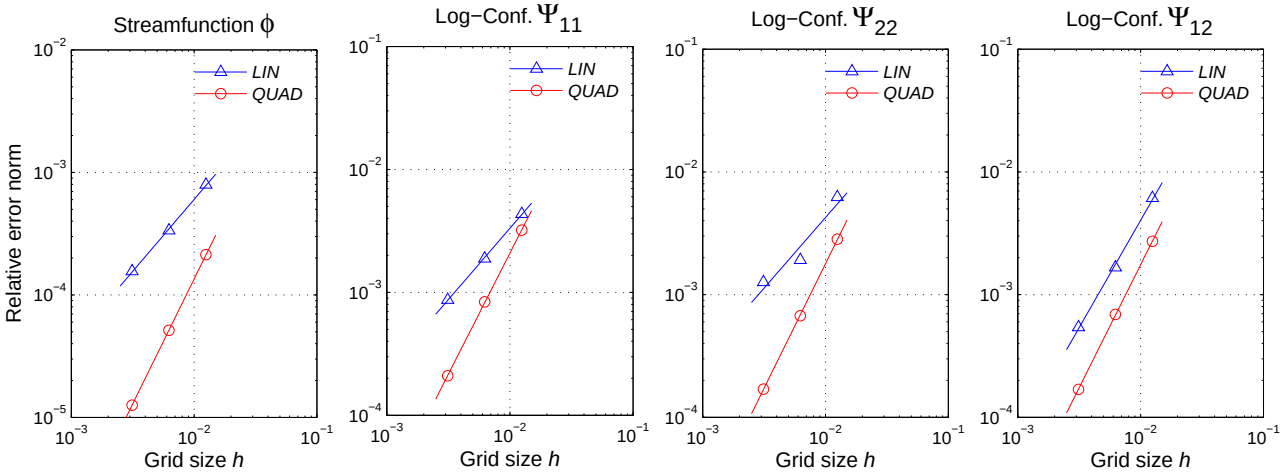


Figure 6.3: Asymptotic convergence of the spatial relative error norm, for the LIN and QUAD stress interpolation schemes.

Variables	Spatial convergence rates	
	LIN interpolation scheme	QUAD interpolation scheme
ϕ	1.17	2.04
Ψ_{11}	1.16	1.97
Ψ_{22}	1.15	2.03
Ψ_{12}	1.75	2.00

Table 6.1: Spatial convergence rates with the LIN and QUAD stress interpolation schemes.

The numerical solution has a spatial convergence rate p when its relative error norm asymptotically converges to the zero grid size limit as:

$$E_r(h) \underset{h\to 0}{\simeq} \kappa h^p + O\left(h^p\right), \tag{6.6}$$

where κ is a constant. The convergence rate is determined as the slope of the best fitting line of the relative errors norm plotted on logarithmic scale, see Figure 6.3, where:

$$\log\left[E_r(h)\right] \simeq \log(\kappa) + p\log(h). \tag{6.7}$$

The spatial convergence rates for the streamfunction and the three components of the log–conformation tensor are reported in Table 6.1, for the LIN and QUAD stress interpolation schemes. As it can be seen, only the QUAD scheme exhibits true second order accuracy. The convergence rate of the LIN scheme—which is used both by Fattal and Kupferman (2005) and Hao and Pan (2007)—is between first and second orders. This confirms that linear stress interpolations, in the momentum equations, degrade the second-order accuracy of the CUBISTA and the central–differencing schemes.

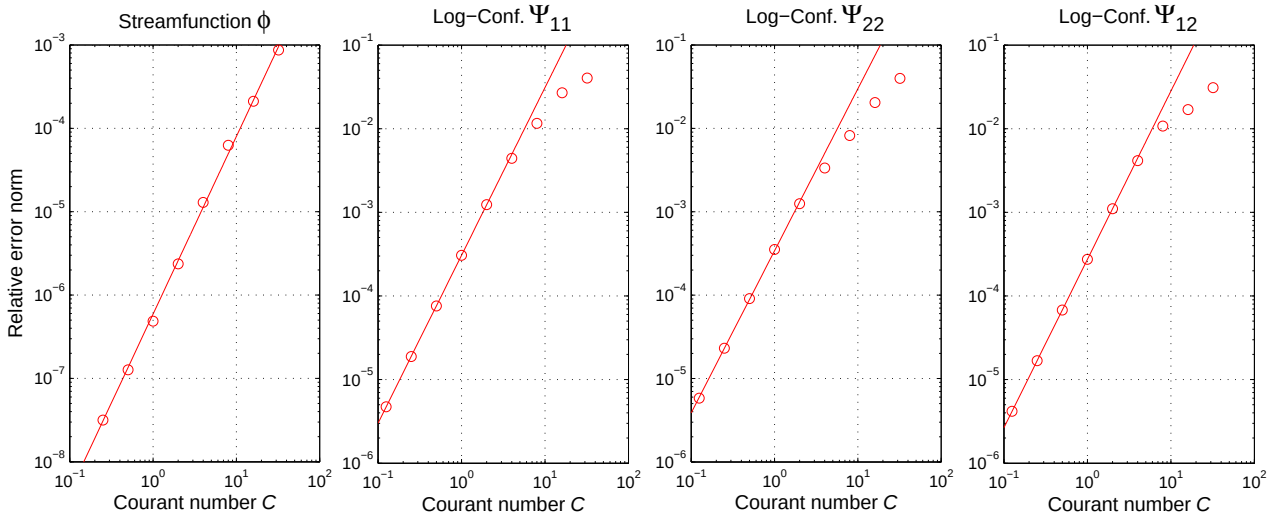


Figure 6.4: Asymptotic convergence of the temporal relative error norm.

Variables	Temporal convergence rates
ϕ	2.13
Ψ_{11}	1.97
Ψ_{22}	1.96
Ψ_{12}	1.97

Table 6.2: Temporal convergence rates, for $C < 8$.

The temporal convergence is assessed from the transient solution at the time $t = 2$, for $Wi = 1$, such that the flow has not reached its steady-state yet. The simulation is performed for different time-step sizes, but with the same grid spacing $h = 1/160$, using the QUAD stress interpolation scheme. The temporal convergence rates are evaluated similarly as for the spatial convergence rates, except that the convergence is calculated with respect to refinements of the time-step size (i.e. the Courant number C), see Figure 6.4. The temporal convergence is of second order accuracy, for Courant numbers below 8. Above this value, the convergence of the log–conformation tensor is degraded to first order, because the time increments become too large to catch the details of the transient flow and the ramping of the lid velocity with the hyperbolic tangent function. The temporal convergence rates, for Courant numbers $C < 8$, are provided in the Table 6.2.

It is interesting to note that stable solutions can be obtained with Courant numbers as large as $C = 64$, in contrast to the decoupled velocity–pressure methods (Xue, Tanner & Phan-Thien 2004). This result demonstrates the robustness of the fully implicit streamfunction method for solving the conservation laws. At large Courant numbers, each time-step requires more iterations in order to converge, but the simulation comprises less time-steps. As we use a direct matrix solver, the computation cost is directly proportional to the total number of iterations within the entire simulation. Figure 6.5(left) compares the recorded computational costs for the different Courant numbers. It clearly shows that the use of large time-step sizes is beneficial, as far as the computational cost is concerned. However, the precision of the calculations should also be considered when assessing the computational efficiency.

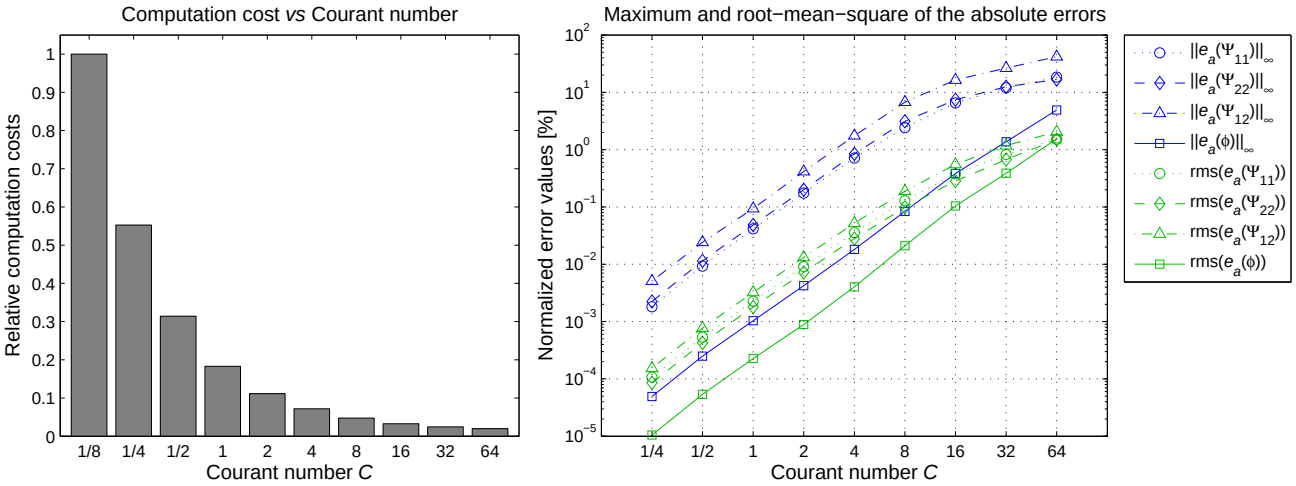


Figure 6.5: Computational efficiency: the relative computation costs (left) and the normalized absolute errors (right), for different Courant numbers.

The absolute errors of each variable are evaluated at all the grid points, and normalized by the largest values:

$$\widehat{e}_a\left(\chi_C, x_i, y_j\right) = \frac{\left|\chi_C\left(x_i, y_j\right) - \tilde{\chi}_0\left(x_i, y_j\right)\right|}{\max\limits_{i,j}\left[\tilde{\chi}_0\left(x_i, y_j\right)\right]}, \tag{6.8}$$

where $\widehat{e}_a\left(\chi_C, x_i, y_j\right)$ are the normalized absolute errors, $\chi_C\left(x_i, y_j\right)$ is the numerical solution obtained with the Courant number C, and $\tilde{\chi}_0$ is the reference solution obtained with the finest time increment ($C = 1/8$). Figure 6.5(right) provides the maximum values and the root-mean-squares of the normalized absolute errors of each variable. These data can be used to estimate the maximum allowable time-step size (i.e. the Courant number), in order to reach a given numerical precision.

6.1.2. Simulations at moderate Weissenberg numbers

In this subsection, the transient viscoelastic lid-driven cavity flow is solved for the moderate Weissenberg numbers $Wi = 1$, 2 and 3. The numerical solutions are compared with the available results in the literature (Fattal & Kupferman 2005; Hao & Pan 2007). A non-uniform Cartesian grid comprising 250×250 control volumes is used. The grid is symmetrically refined near the walls, with a constant contraction ratio $\Delta x_{k+1}/\Delta x_k = 0.96$ when approaching the walls, see Figure 6.6. The maximum and minimum grid spacings are 0.02 at the center of the cavity, and 1.27×10^{-4} at the walls, respectively. The computational efficiency is enhanced with an adaptive time-stepping procedure described in section 4.4.

The solution of the streamlines and the components of the log–conformation tensor are represented in Figures 6.7, 6.9 and 6.11, at $t = 8$ for $Wi = 1$, and $t = 40$ for $Wi = 2$ and 3, respectively. Figures 6.8, 6.10 and 6.12 provide velocity and log–conformation profiles along the vertical line $x = 1/2$ and horizontal line $y = 3/4$, for $Wi = 1$, 2 and 3, respectively. These figures also display the history of the specific kinetic energy E_k, and the specific elastic energy E_e (i.e. the Helmholtz free energy), defined as:

$$E_\mathrm{k} = \frac{1}{2}\int_\Omega \left(u^2 + v^2\right) \mathrm{d}V, \tag{6.9}$$

$$E_e = \frac{1}{2}\int_\Omega \mathrm{tr}\left(\boldsymbol{c} - \boldsymbol{I}\right)\mathrm{d}V \,. \tag{6.10}$$

For $Wi = 1$ and 2, the results are in good agreements with those of Fattal and Kupferman (2005) and Hao and Pan (2007). For $Wi = 3$, some disparities of the results are observed in the vicinity of the upper right corner. However, the present solution is obtained with grid spacings near the walls 30 times finer than in Fattal and Kupferman (2005), and a second-order accurate scheme, which give us confidence in our results. As observed by Fattal and Kupferman (2005), the solution for $Wi = 3$ exhibits small oscillations due to vortices repeatedly created and annihilated, in the vicinity of the upper right corner. Moreover, Fattal and Kupferman (2005) reported convergence difficulties attributed to the under-resolution of subgrid vortices. Under-resolution is generally manifested by chaotic oscillations in the history of the kinetic energy or the elastic energy. However, in the present simulation the history of the kinetic and elastic energies exhibits a quasi-periodicity, for $Wi \geq 2$. This quasi-periodicity underpins that the oscillations in the flow are likely to be linked to a structural pattern of elastic instabilities rather than numerical errors. A further discussion on the elastic instabilities is given in the next subsection.

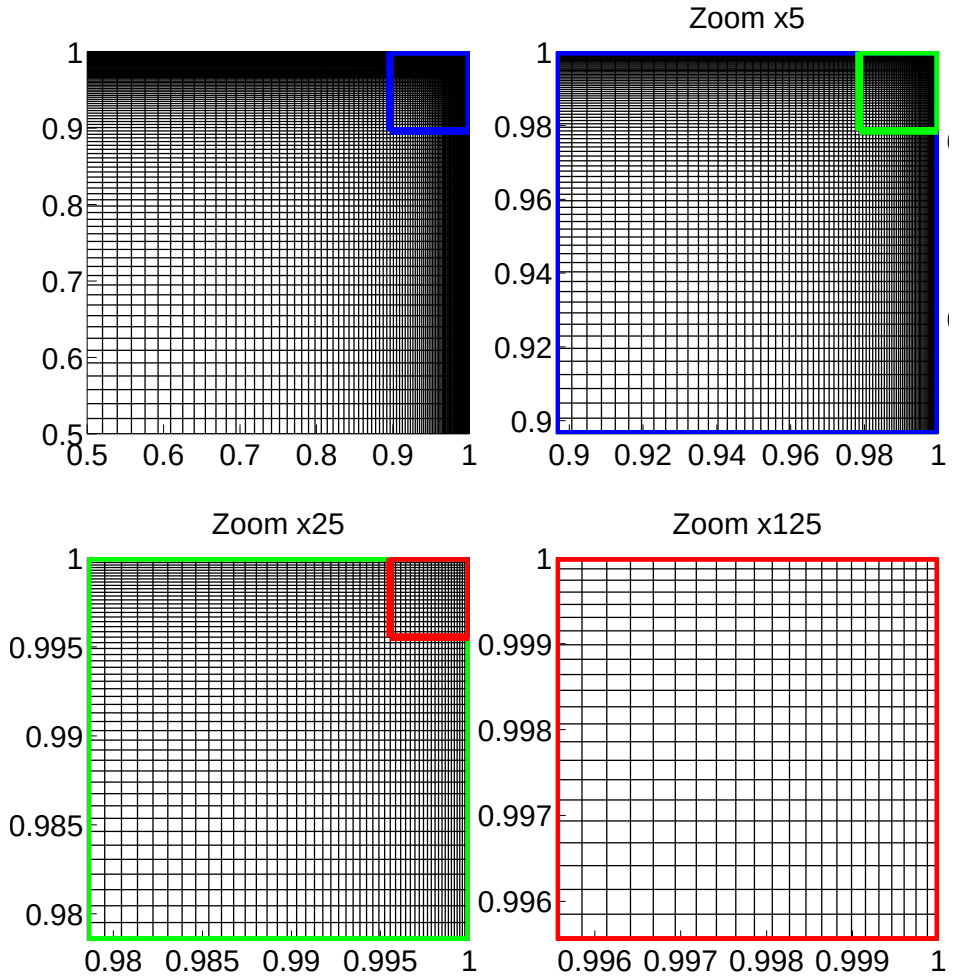


Figure 6.6: Zooms on the upper right corner of the grid having a constant contraction ratio $\Delta x_{k+1}/\Delta x_k = 0.96$.

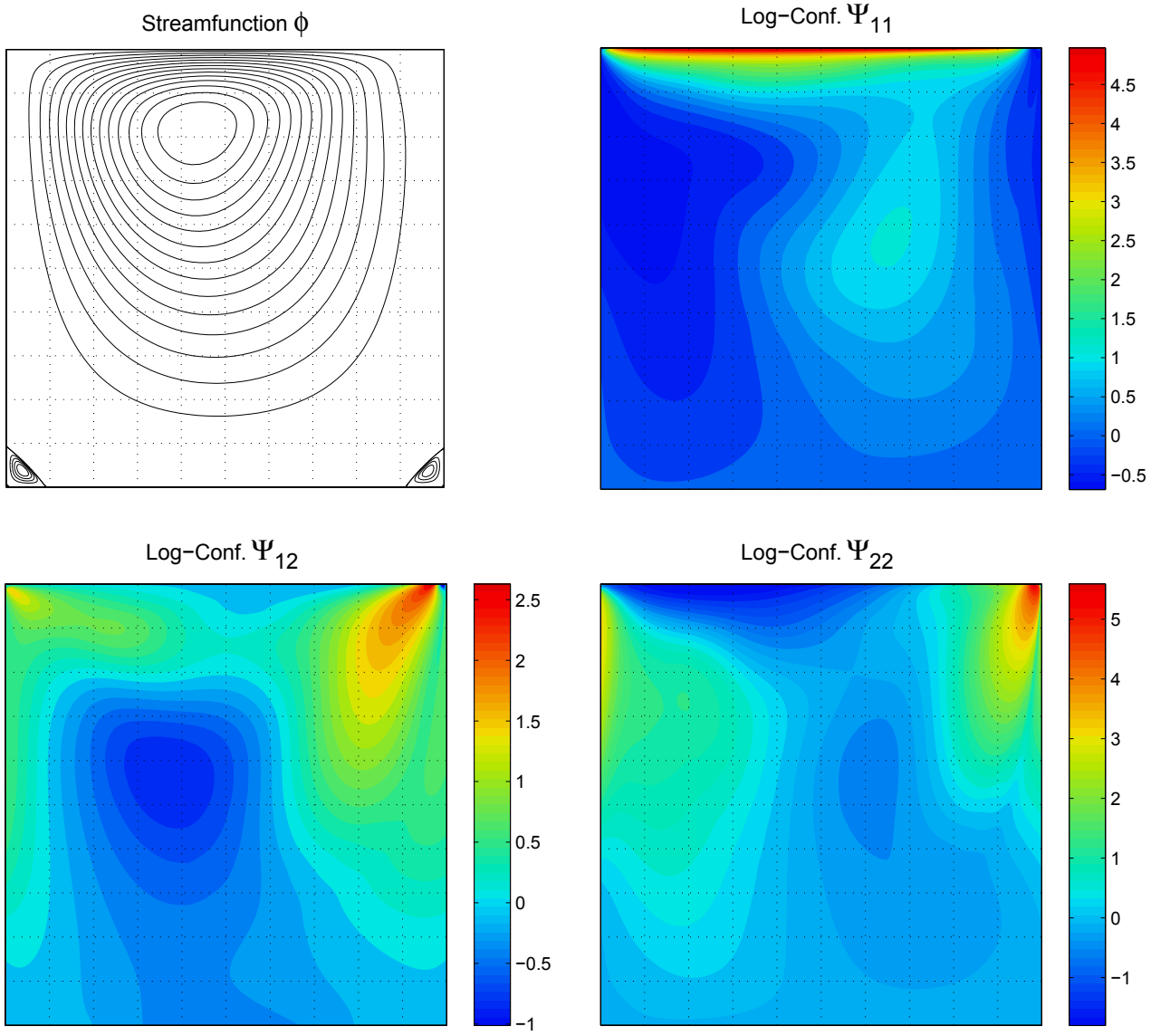


Figure 6.7: The streamlines and the three components of the log–conformation tensor, at time $t = 8$, for $Wi = 1$. (The extrema of the color scales do not correspond to the extrema of the solution.)

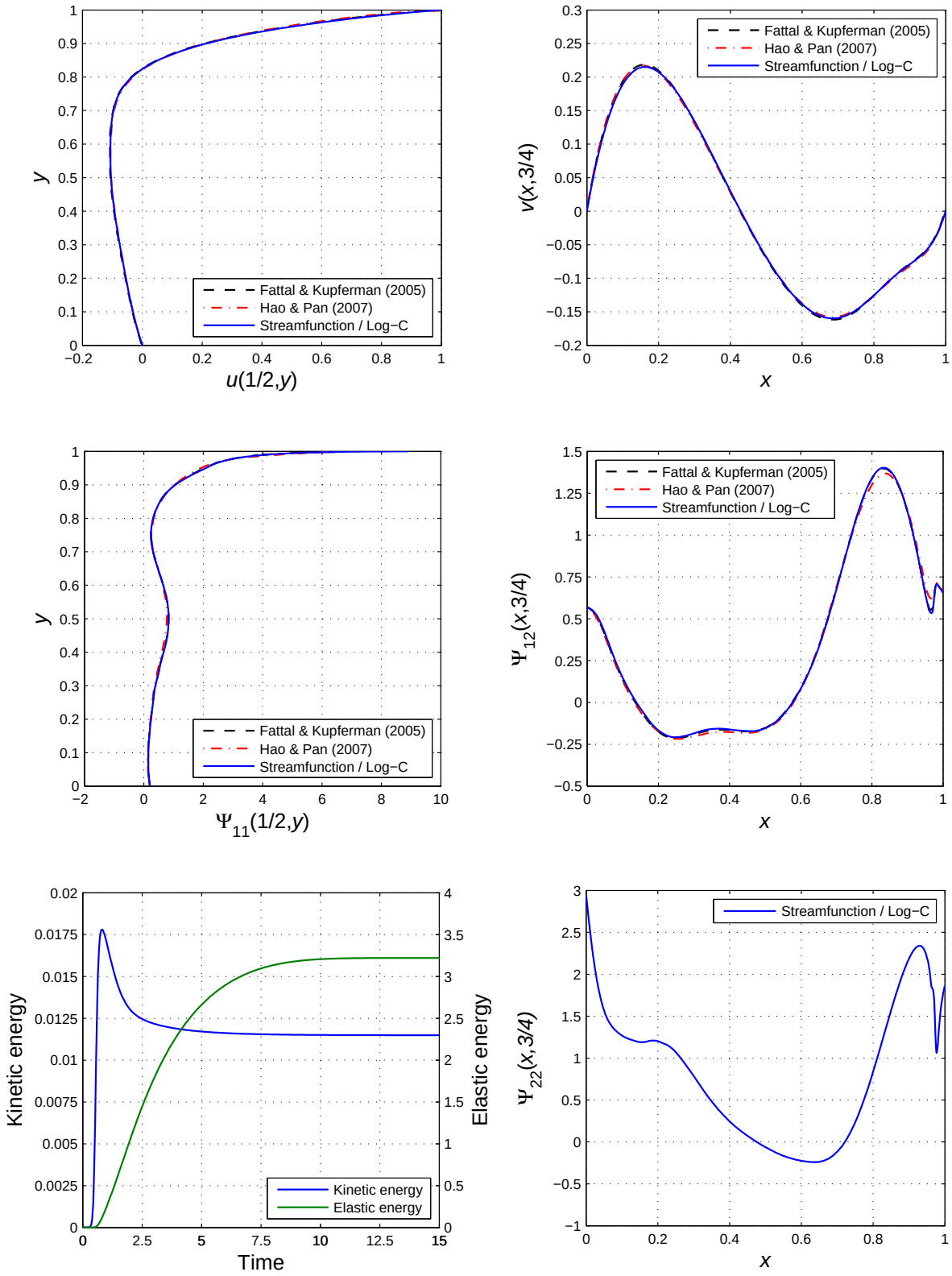


Figure 6.8: Results at time $t = 8$, for $Wi = 1$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$; history of the kinetic energy and elastic energy.

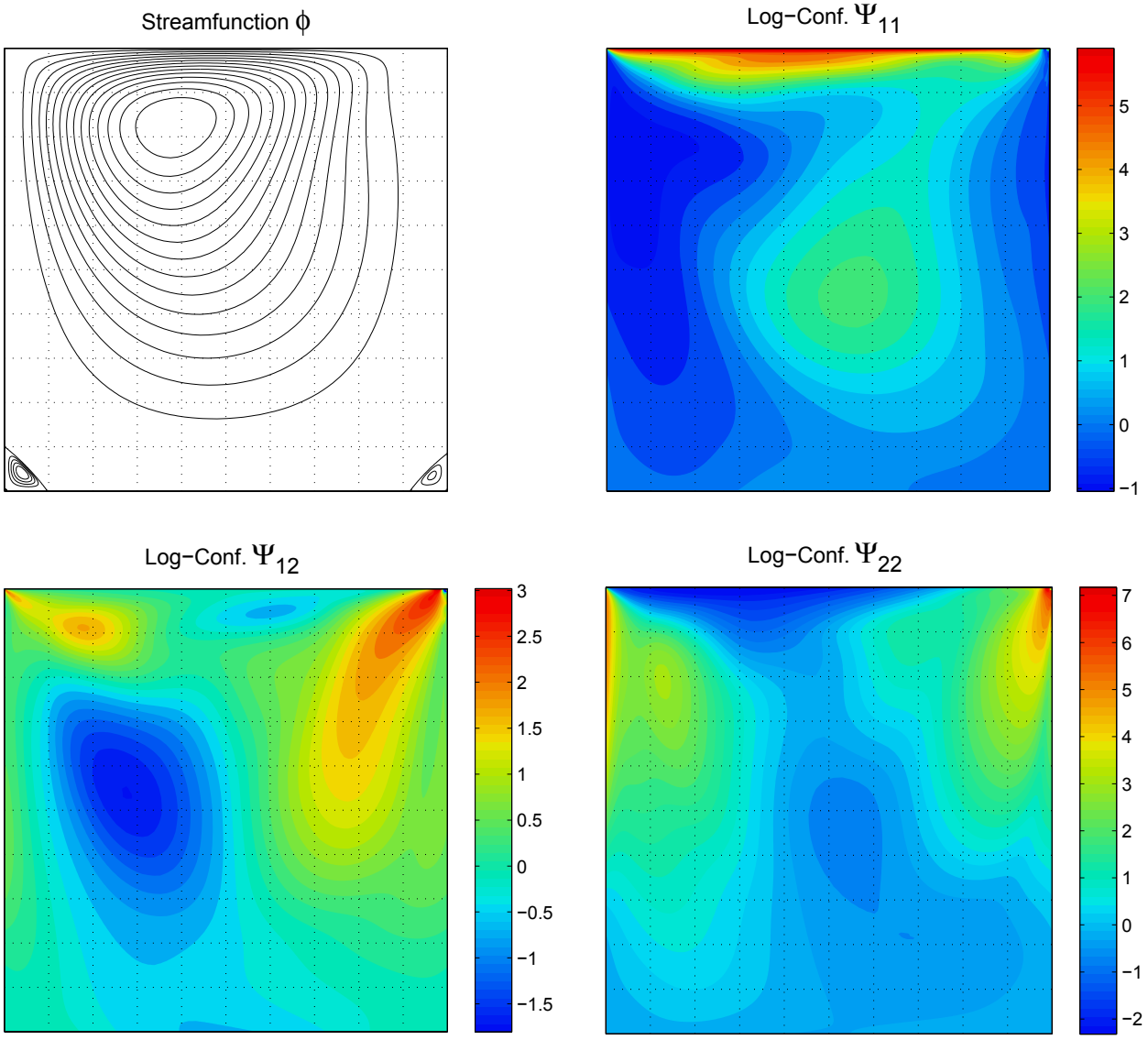


Figure 6.9: The streamlines and the three components of the log–conformation tensor, at time $t = 40$, for $Wi = 2$. (The extrema of the color scales do not correspond to the extrema of the solution.)

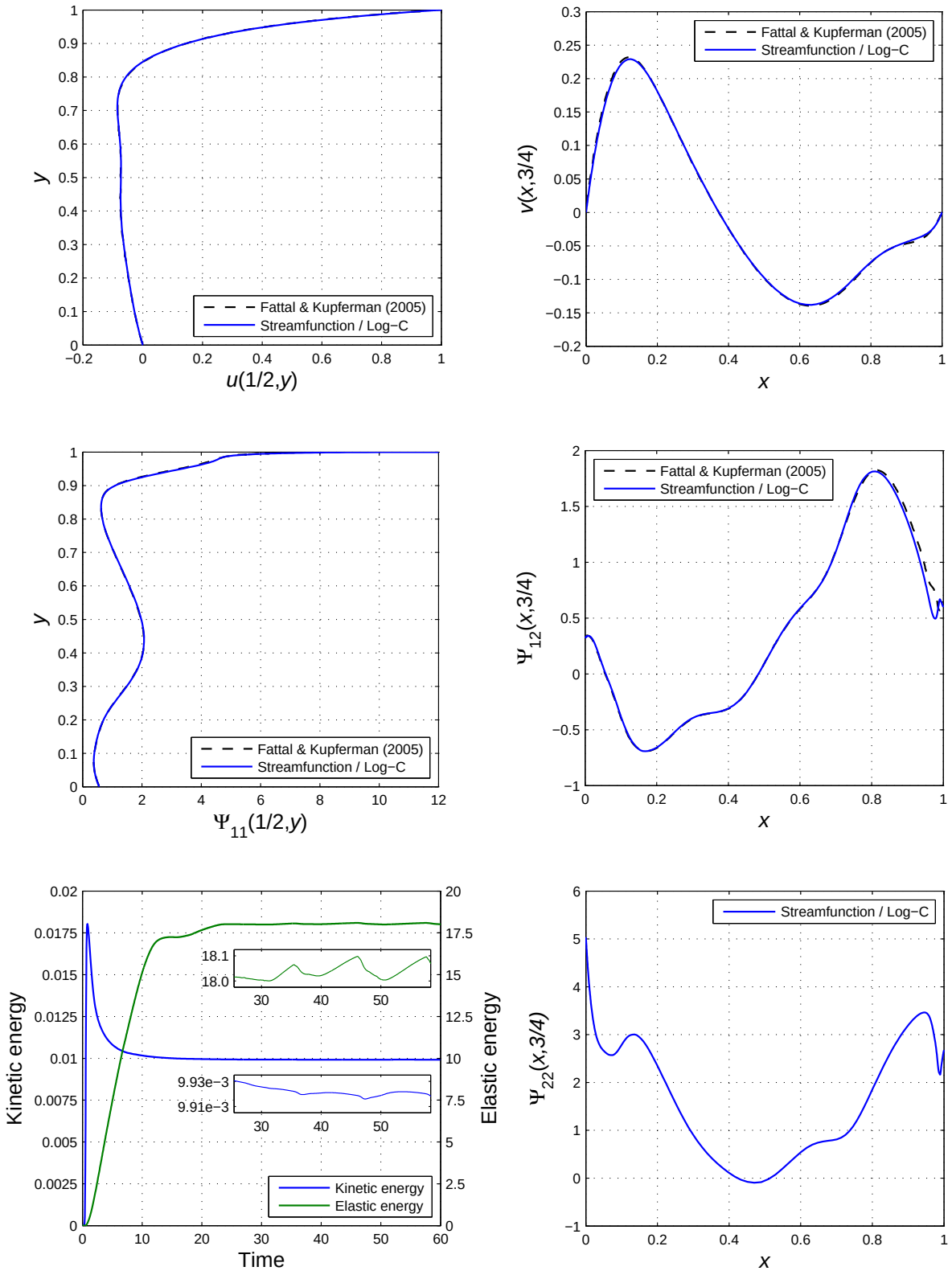


Figure 6.10: Results at time $t = 40$, for $Wi = 2$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$; history of the kinetic energy and elastic energy.

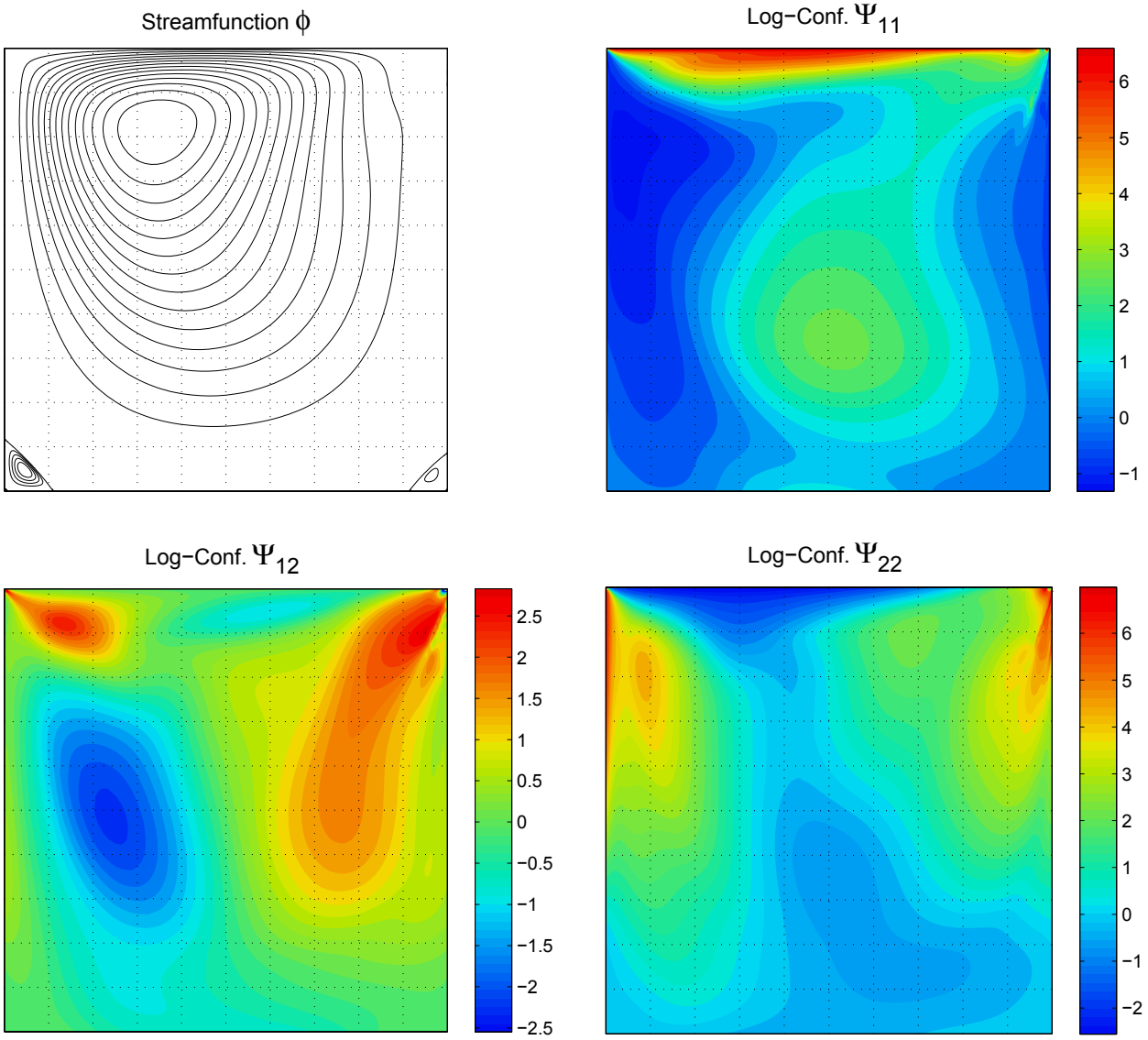


Figure 6.11: The streamlines and the three components of the log–conformation tensor, at time $t = 40$, for $Wi = 3$. (The extrema of the color scales do not correspond to the extrema of the solution.)

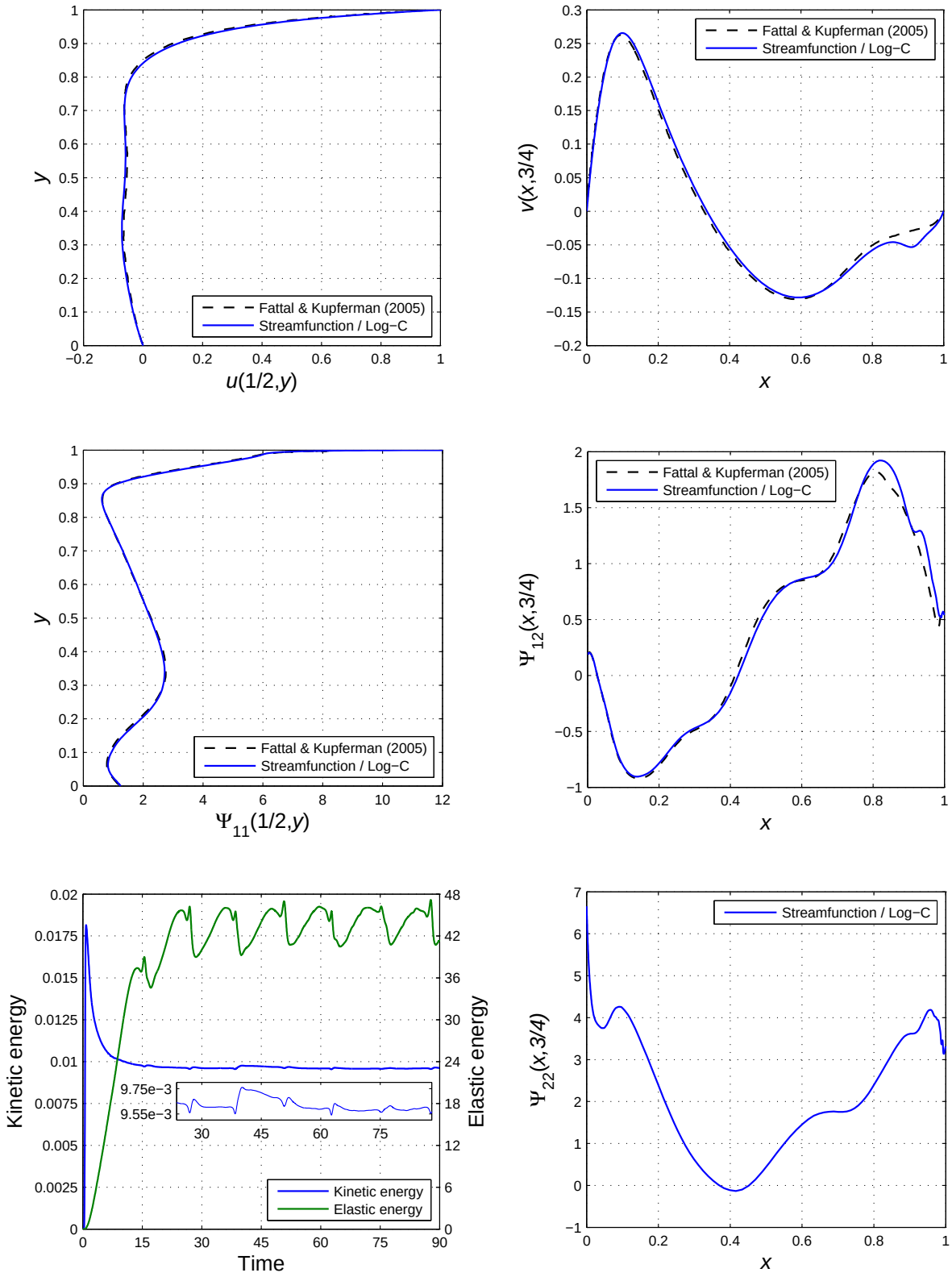


Figure 6.12: Results at time $t = 40$, for $Wi = 3$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$; history of the kinetic energy and elastic energy.

6.1.3. Simulations at high Weissenberg numbers

To the knowledge of the author, numerical results of the lid-driven cavity flows of an Oldroyd–B liquid at the Weissenberg numbers higher than 3 have never been reported in the literature. This subsection presents new results for the lid-driven cavity flow, at the high Weissenberg numbers $Wi = 5$ and 10. The same mesh and the same adaptive time-stepping procedure are used as in the previous simulations for lower Weissenberg numbers. The results for $Wi = 5$ and 10 present similar features that are described below.

After the lid velocity has reached its maximum value, the flows undergo transient developments, where the kinetic energy slowly decreases, while elastic energy gets stored in the fluid. The transient developments of the flows take longer time at high Weissenberg numbers, because the fluid has larger relaxation time. During the flow development, some vortices appear at the upper right corner and move downward along the wall, until they either annihilate or merge with the vortex at the lower corner. At some point, the flows evolve into quasi-periodic unsteady states. The quasi-periodicity of the flows are clearly visible in the histories of the specific elastic and kinetic energy, represented in Figures 6.13 and 6.14 for $Wi = 5$ and 10, respectively. The amplitude of the oscillations increases with the Weissenberg number, but not the quasi-periodicity, interestingly. Figures 6.13 and 6.14 also display the evolution of the center of the primary vortex, as well as the normalized intensity of vortices generated at the upper corners (scaled by the intensity of the primary vortex). The red and yellow colored areas on the graphs correspond to the time-intervals of noticeable secondary vortices at the upper left and right corners, respectively. The results show a clear correlation between the appearance of secondary vortices at the upper left corner and the flows' quasi-periodicity.

When examining the region of the upper left corner in detail, the presence of a small corner vortex can be seen in Figures 6.15 and 6.16, for $Wi = 5$ and 10, respectively. However, the corner vortex has such a low intensity that it can be consisted as virtually stagnating, within the time scale of the material relaxation. Nevertheless, the fluid at the interface between the corner vortex and the primary vortex is subjected to shear deformations. Thus, an elastic stress layer is built-up at the interface between the vortices, as it can be seen in Figures 6.15 and 6.16, where the stress level is monitored by the entrywise 2-norm of the log-conformation tensor $\|\boldsymbol{\Psi}\|_2 = \sqrt{\sum_{ij} \Psi_{ij}^2}$. The large elastic stress stored in this stress layer is responsible for the increases of elastic energy observed in Figures 6.13 and 6.14. When the accumulated elastic energy reaches a threshold value, another secondary vortex appears just below the corner vortex. The secondary vortex destroys the elastic stress layer while it grows and propagates downward the wall. The elastic energy stored in the stress layer is partly converted into kinetic energy and partly dissipated through viscous dissipation. The secondary vortex also affects the primary vortex of the cavity. Its center gets pushed to the right and starts to oscillate, which results in the fluctuation in the kinetic and elastic energy, see Figure 6.13 and 6.14. Finally, the secondary vortex annihilates when most of the energy elastic stored in the corner region is cleared, and the system finds itself at the beginning of another cycle, with the formation of a new corner vortex. To the knowledge of the author, this structural mechanism of the elastic instabilities in upper left corner of the lid-driven cavity has never been identified in numerical simulations before. It is interesting to note that the elastic instabilities are solely initiated at the corner, where the flow has curved streamlines, while the elastic stress layer is also built-up along the left wall and the moving lid. This feature is consistent with the general mechanism of elastic instability in curved streamlines, discussed by Pakdel and McKinley (1996).

Small eddies are also generated and annihilated at the upper right corner of the cavity, see Figure 6.17. However, only few of them develop a reasonable size, and they globally have no noticeable impact on the primary flow. Moreover, they do not present any regular pattern, and their occurrence seems independent from the quasi-periodic oscillations of the rest of the flow. Aperiodic fluctuations of the viscoelastic flow at the upper right corner of the lid-driven cavity have been observed experientially by Pakdel and McKinley (1998). Whether these fluctuations originate from elastic turbulence is still an open question.

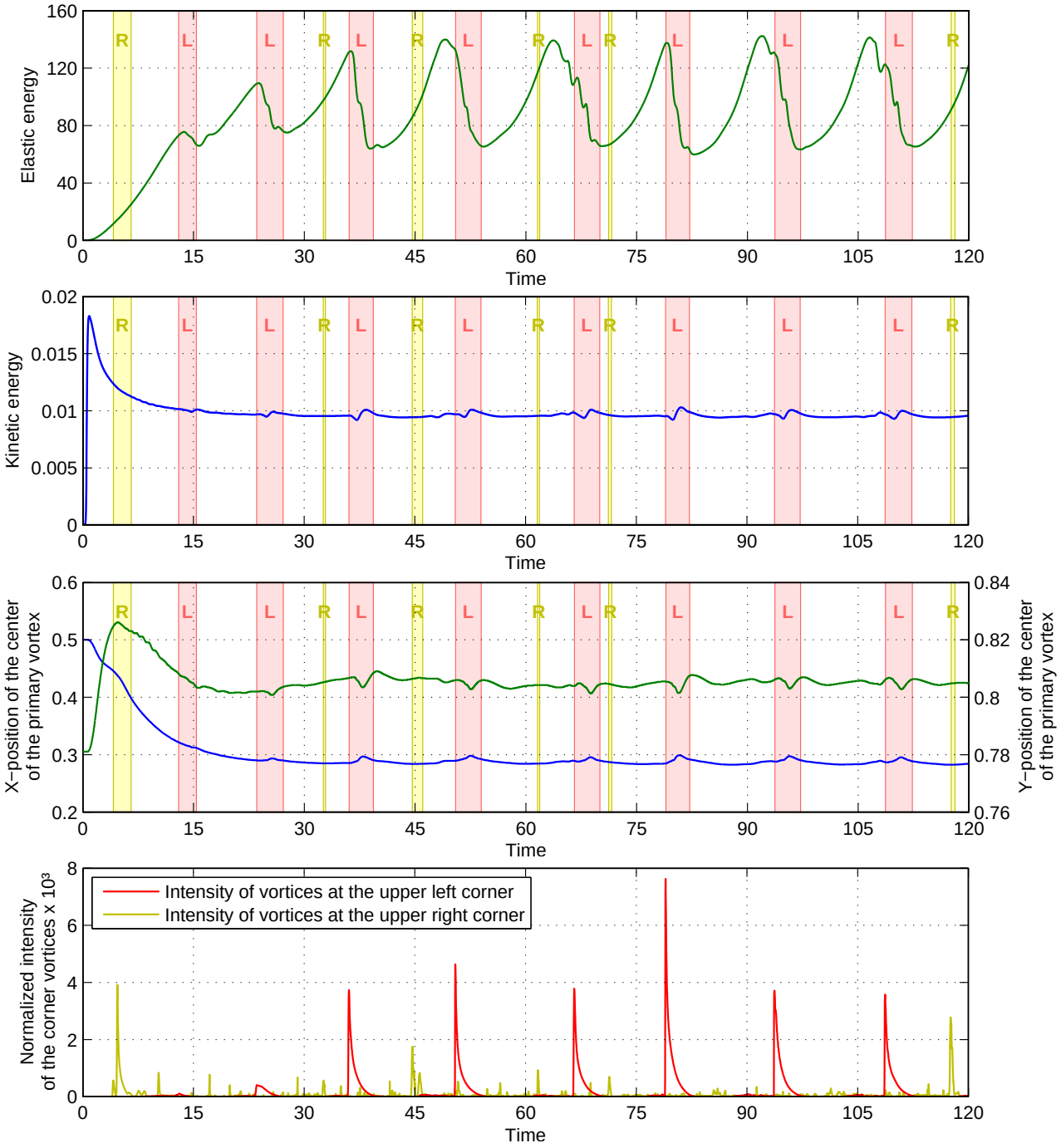


Figure 6.13: History of the elastic and kinetic energy, evolution of the center of the primary vortex, and intensity of secondary vortices generated at the upper corners, for $Wi = 5$. The red and yellow colored areas correspond to the time-intervals of noticeable secondary vortices, at the upper left and right corners, respectively.

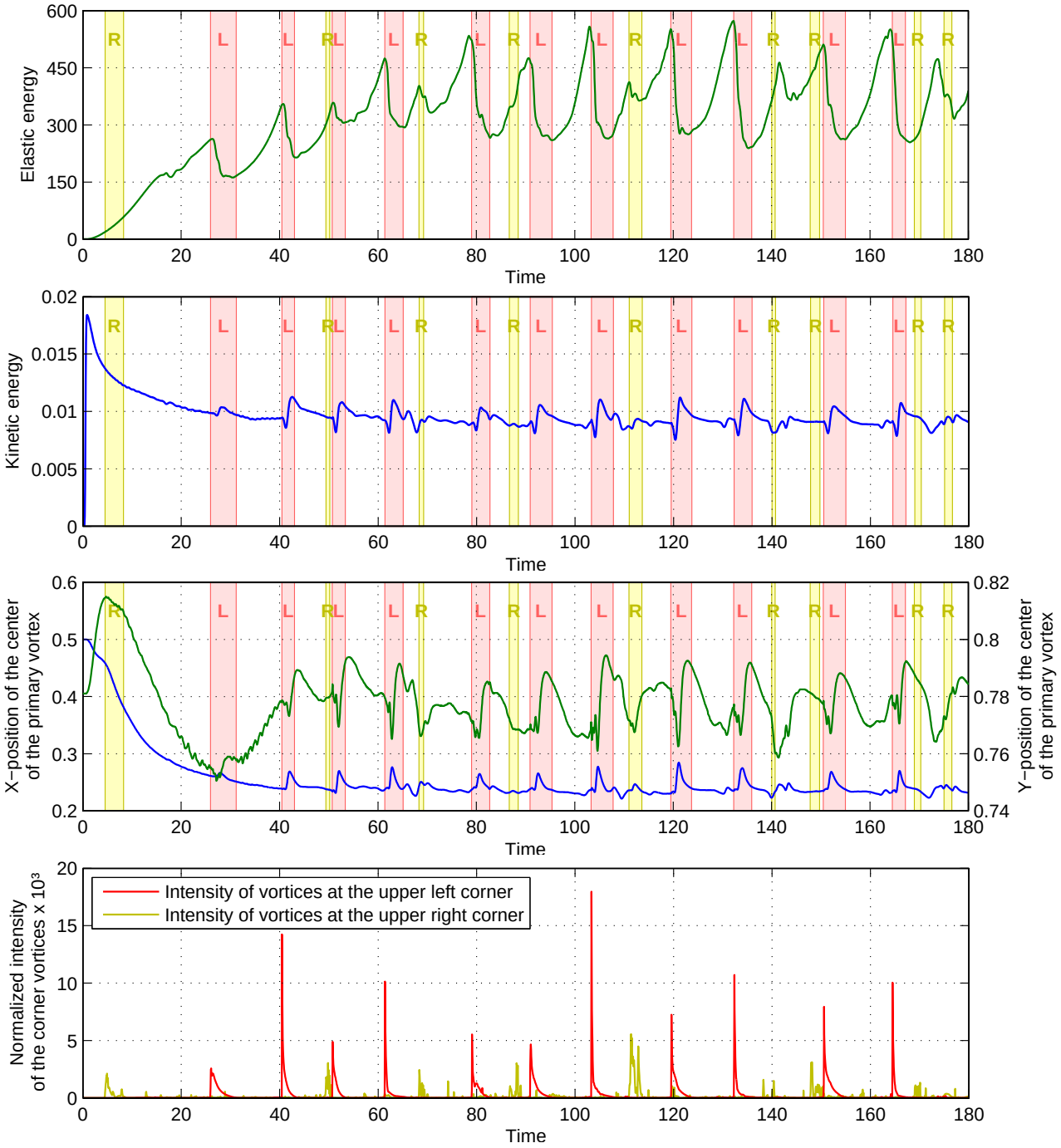


Figure 6.14: History of the elastic and kinetic energy, evolution of the center of the primary vortex, and intensity of secondary vortices generated at the upper corners, for $Wi = 10$. The red and yellow colored areas correspond to the time-intervals of noticeable secondary vortices, at the upper left and right corners, respectively.

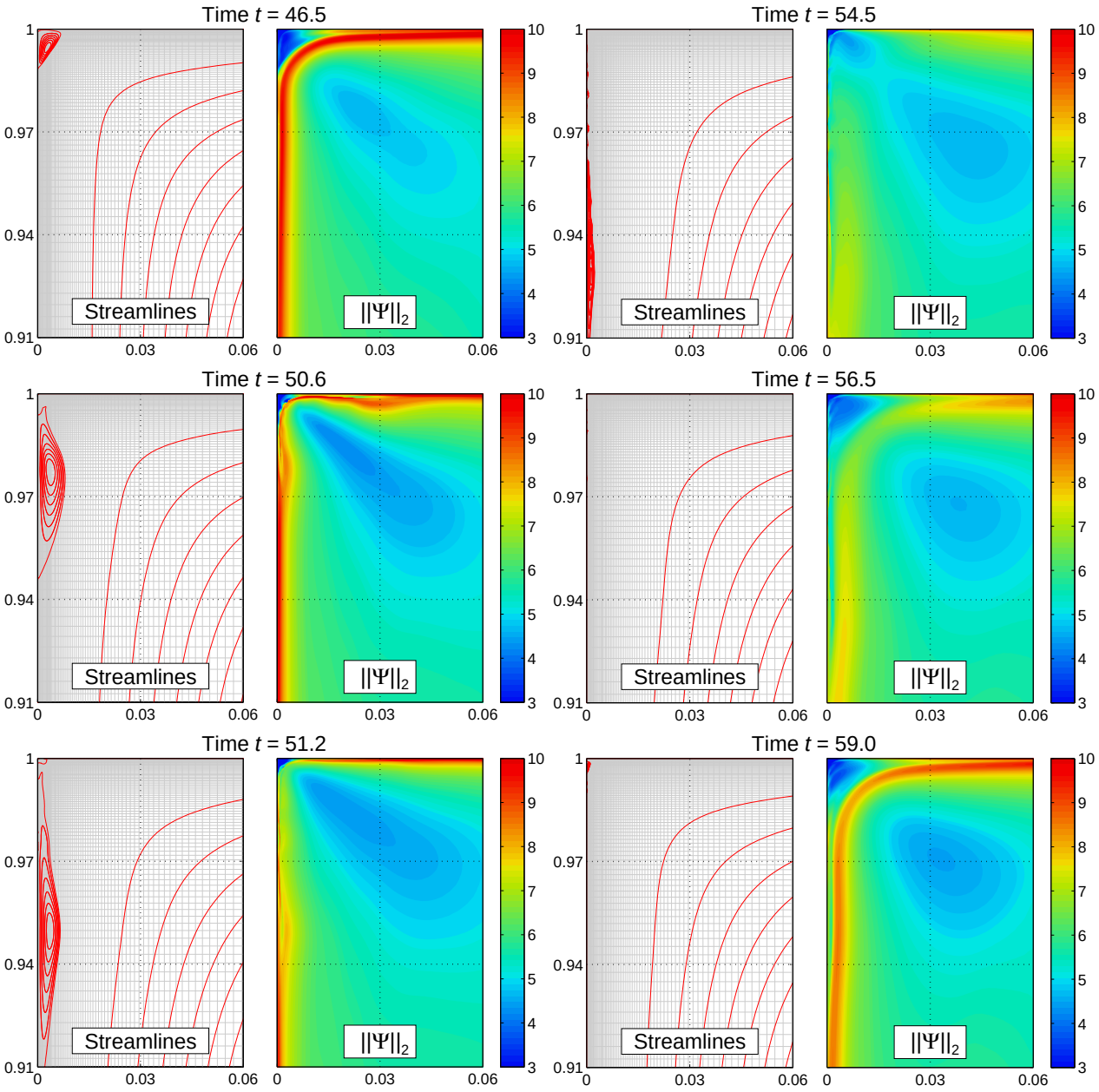


Figure 6.15: Snapshots of the streamlines and the 2-norm of the log–conformation tensor at the upper left corner, for $Wi = 5$.

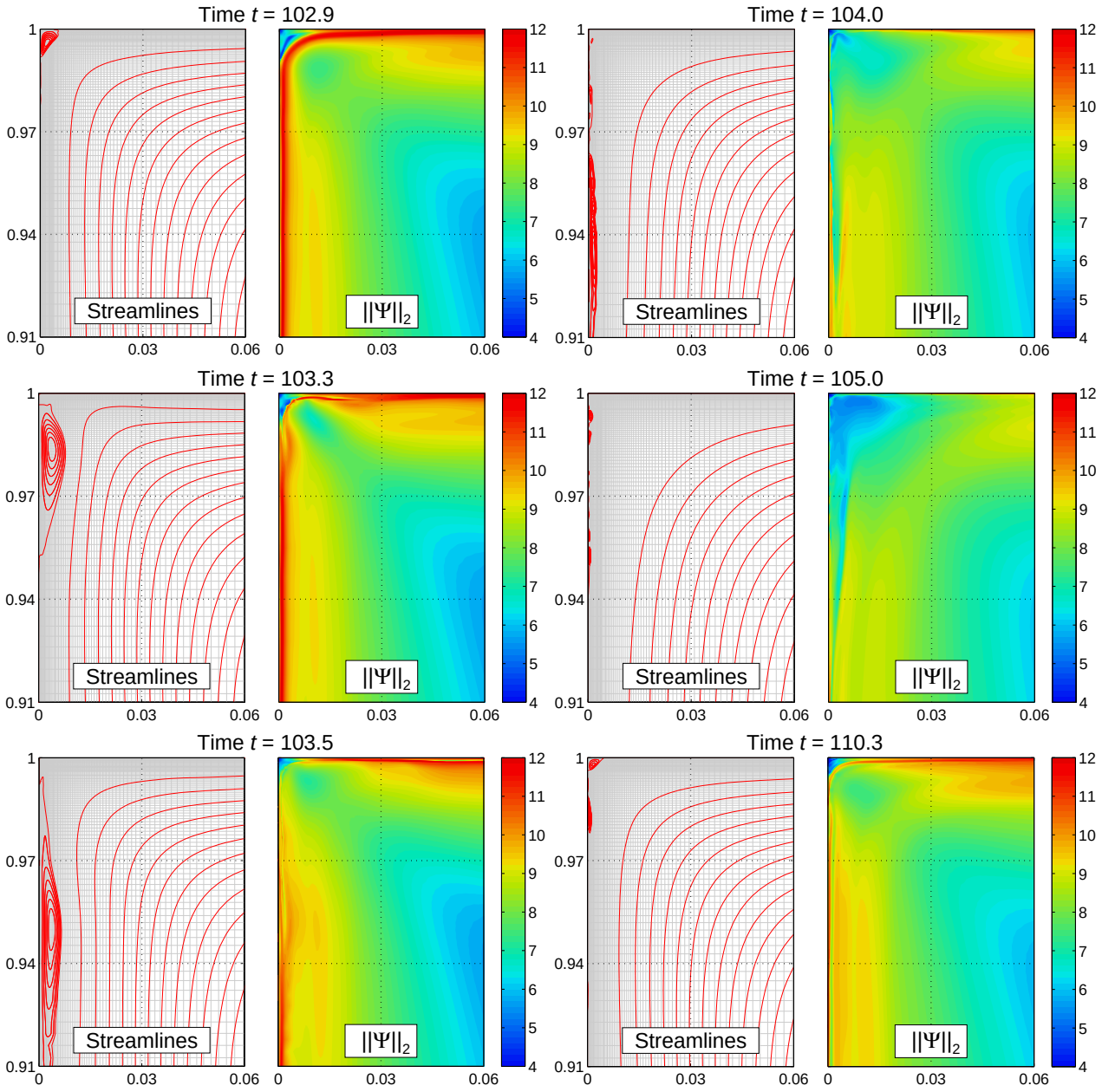


Figure 6.16: Snapshots of the streamlines and the 2-norm of the log–conformation tensor at the upper left corner, for $Wi = 10$.

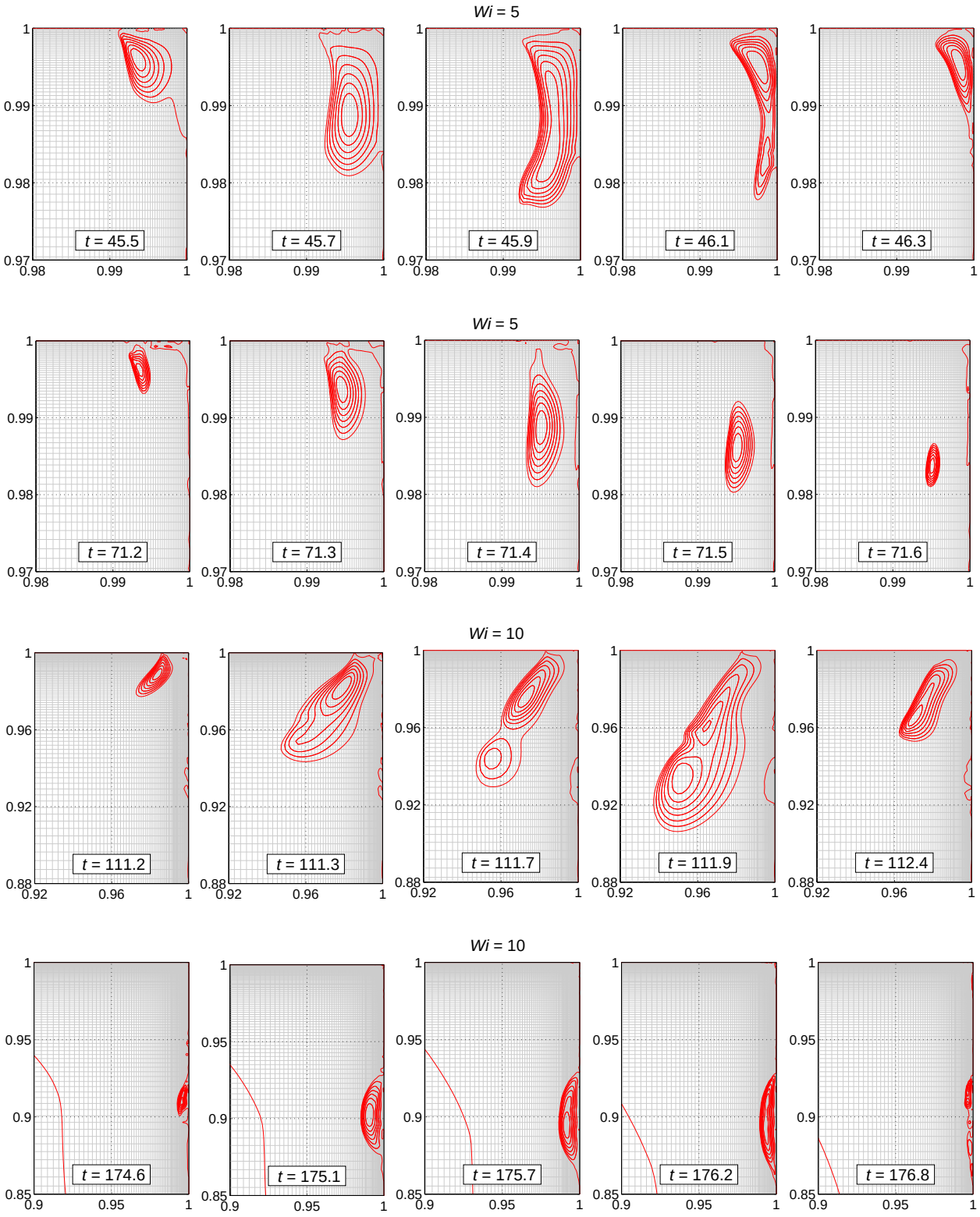


Figure 6.17: Specimens of secondary vortex generated at the upper right corner, for $Wi = 5$ and 10 .

6.2. Single phase viscoelastic flows through a 4:1 planar contraction

The 4:1 planar contraction is probably one of the most popular benchmark tests for viscoelastic flow solvers. The contraction flows of viscoelastic fluids have been the subject of many experimental investigations, in planar, axisymmetric and three-dimensional geometries (Boger 1987; McKinley et al. 1991; Yesilata, Öztekin & Neti 1999; Sousa et al. 2009; Sousa et al. 2011; Rodd et al. 2010). Moreover, as pointed out in the Report on the Vllth International Workshop on Numerical Methods in Viscoelastic Flows (Brown & McKinley 1994), it is necessary for computational rheologists to have common benchmark tests in order to perform comparative evaluations of the various numerical techniques. A list of selected recent publications addressing the simulation of viscoelastic flows in the 4:1 planar contraction is provided hereinafter.

The accuracy of various finite–element techniques and a hybrid FV/FE method has been investigated by Webster et al. (Aboubacar & Webster 2001; Aboubacar & Webster 2003; Belblidia et al. 2007; Belblidia, Matallah & Webster 2008). Kim et al. (2005) implemented a high-resolution DEVSS-G/DG finite–element algorithm with a parallel multi-frontal solver. Another second-order accurate finite–element code with adaptive mesh refinement near the corner has been developed by Cai and Westphal (2009). Finite–element simulations of the viscoelastic flows in planar contraction, using the log–conformation representation, have been done by Kwon (2004; 2012).

Concerning the finite–volume method, extensive investigations on the different interpolation schemes have been done by Alves et al. (2000; 2003a; 2003b; 2004; Afonso et al. 2011). Edussuriya, Williams and Bailey (2004) used the QUICK scheme to interpolate the flux of the viscoelastic extra–stress. Phillips and Williams (1999; 2002) have implemented a stable semi–Lagrangian advection scheme for the viscoelastic extra–stress. The importance of using high-resolution schemes has been proven by Alves, Pinho and Oliveira (2000), and the CUBISTA interpolation scheme was proposed to enhance the accuracy and the numerical convergence of viscoelastic finite–volume algorithms (Alves, Oliveira & Pinho 2003a). Moreover, the use of the log–conformation representation in the framework of the finite–volume method (Afonso et al. 2009) makes it possible to investigate the dynamics of the entry flows at high-Deborah numbers (Afonso et al. 2011).

The geometry of the 4:1 planar contraction used in the present simulations consist in a reservoir with length $L = 40h$ and width $2H = 8h$, where h is half of the width of the outlet channel. The length of the outlet channel is $l = 20h$, see Figure 6.18. Details on the mesh near the reentrant corner of the abrupt contraction are shown in Figure 6.19. Note that the computational domain includes the entire geometry, since viscoelasticity is responsible for the break-up of the symmetry.

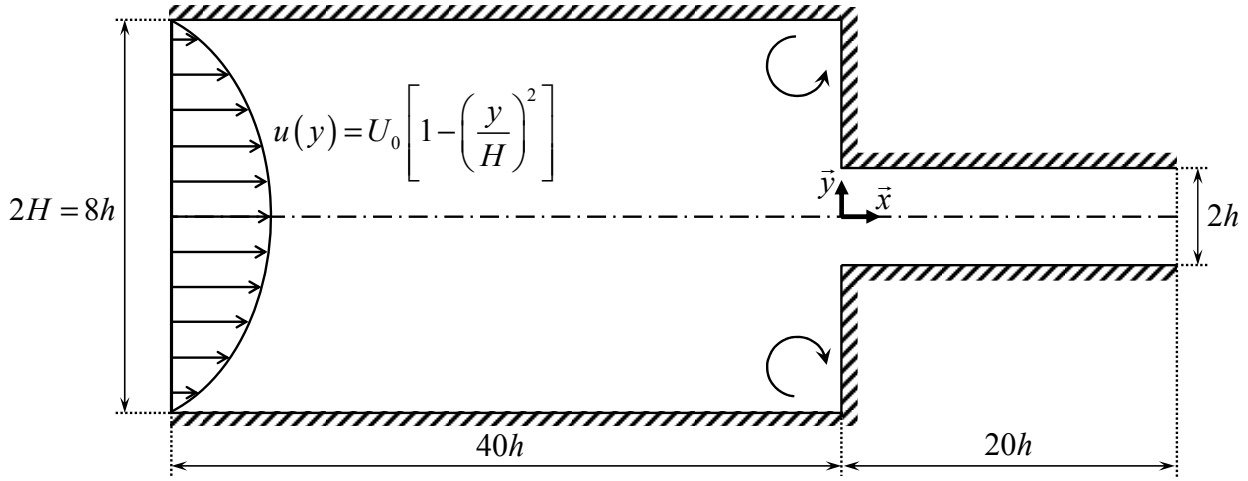


Figure 6.18: Geometry of the 4:1 planar contraction.

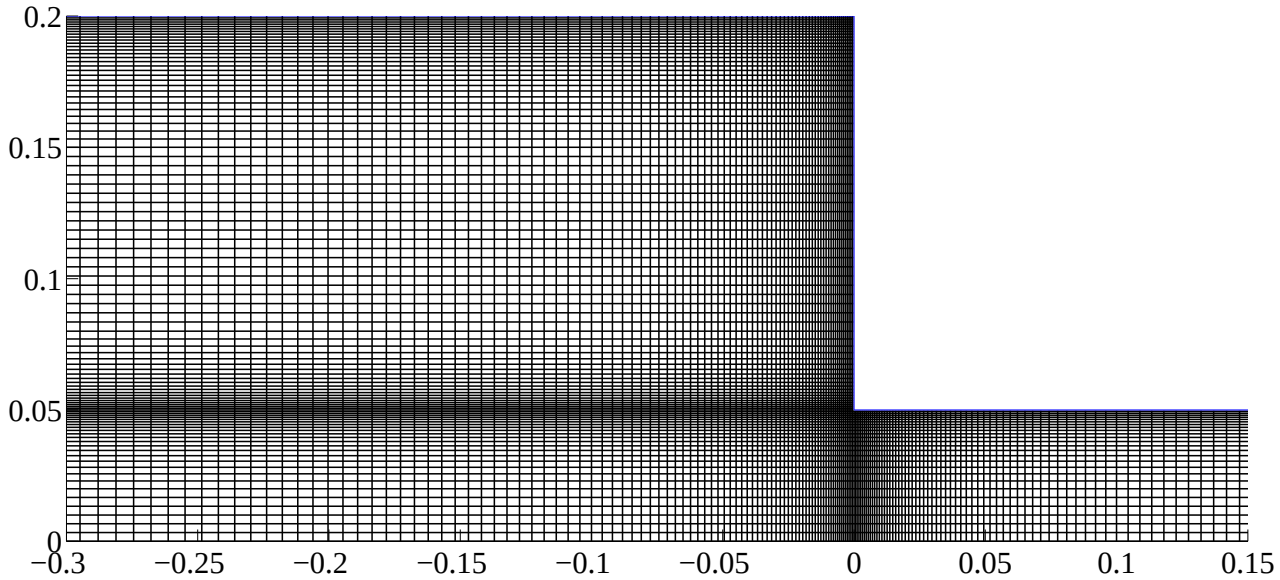


Figure 6.19: Details of the computational mesh near the abrupt contraction (only the upper part is represented).

The no-slip boundary condition is applied at the walls. The parabolic velocity profile of the fully-developed Newtonian creeping flow is specified at the inlet:

$$u_{\text{inlet}}(y) = U_0\left[1-\left(\frac{y}{H}\right)^2\right], \tag{6.11}$$

where U_0 is the maximum of the inlet velocity profile, at the central line. Zero viscoelastic extra-stresses are prescribed at the inlet. Hence, the flow experiences a rearrangement near the inlet, in order to acquire its fully-developed velocity and viscoelastic extra-stress profiles. However, this flow rearrangement near the inlet does not affect the flow in the abrupt contraction, because of the length of the reservoir. At the outlet boundary, the streamfunction is constrained with a zero normal gradient (i.e. the vertical velocity is zero). Consequently, the outlet boundary condition for the viscoelastic extra-stress is a constant extra-stress flux across the outlet.

The Weissenberg number of the viscoelastic flow in the planar contraction is defined as $Wi = \lambda U_{\text{out}}/h$ where $U_{\text{out}} = 8U_0/3$ is the average velocity in the outlet channel. The Reynolds number is defined as $Re = \rho U_{\text{out}} h/\eta_0$. The simulations are performed at the very low Reynolds number $Re = 2.5\times10^{-6}$, such that the inertia terms can be neglected. The retardation ratio of the Oldroyd–B fluid is taken as $\beta = 1/9$, in accordance with the previous studies in the literature. The simulations are performed for various Weissenberg numbers ranging from $Wi = 0$ (Newtonian fluid) to $Wi = 20$. The final time T_f of the simulations are taken as 10 times the relaxation time λ of the Oldroyd–B fluids. Snapshots of the streamlines of the flows, at the end of the simulations, are displayed in Figure 6.20, for various Weissenberg numbers.

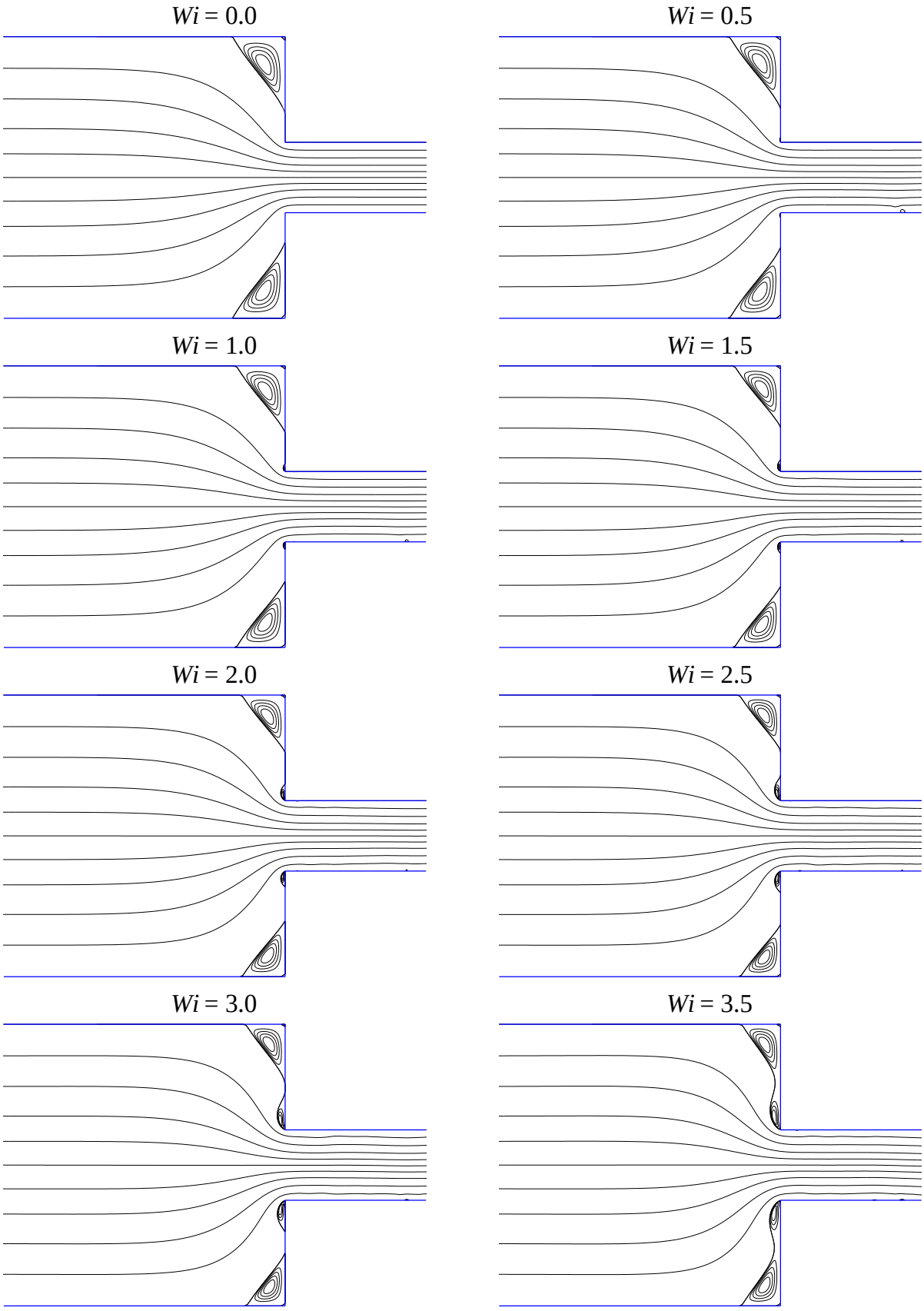


Figure 6.20: Streamlines of the viscoelastic flows in the 4:1 planar contraction, at the end of the simulations.

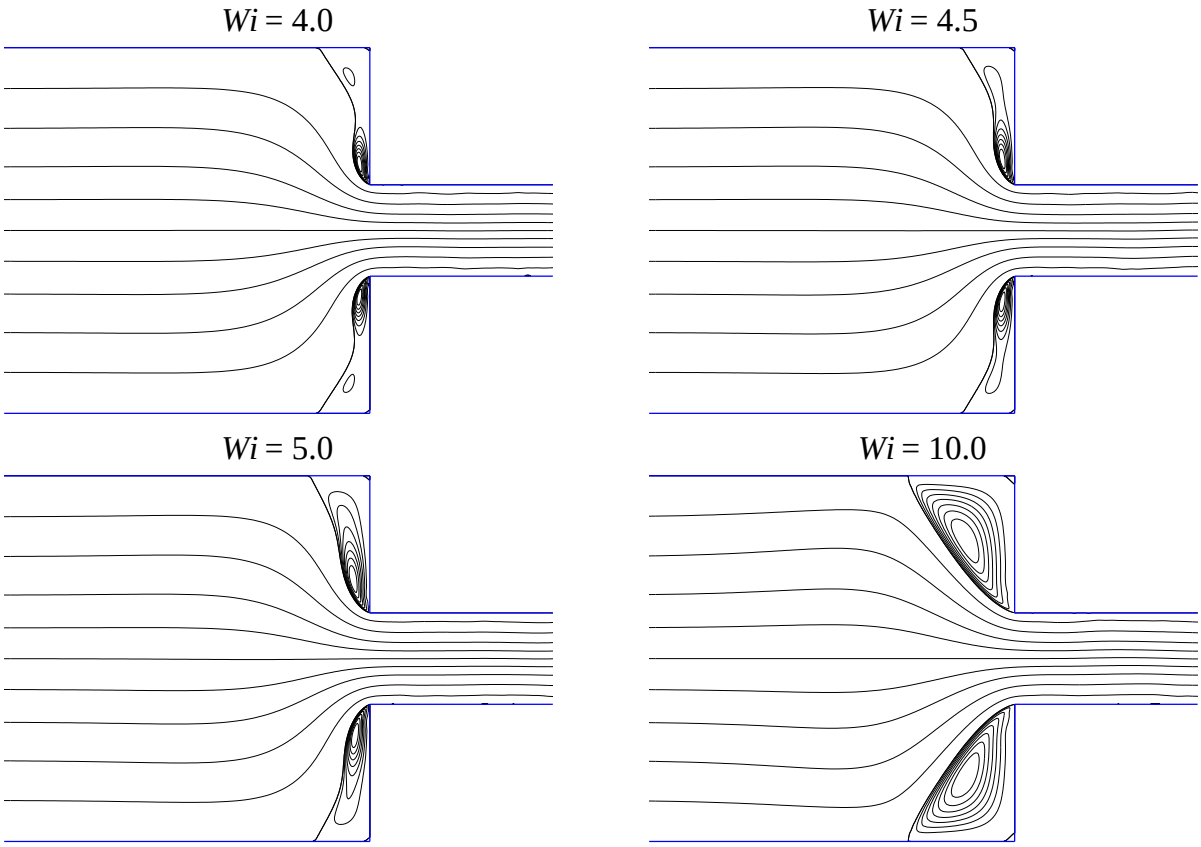


Figure 6.20 (*Continued*)

The flow patterns in Figure 6.20 are in qualitative agreements with the results reported in the literature (Alves, Oliveira & Pinho 2003a; Afonso et al. 2011). The size of the primary vortex at the corners of the reservoir varies as the Weissenberg number increases. A small lip vortex—which is absent in the case of a Newtonian fluid—appears for viscoelastic flows with Weissenberg numbers as low as $Wi = 0.5$. The size of the lip vortex is enhanced as the Weissenberg number increases. At $Wi = 3.5$, the lip vortex starts merging with the corner vortex, but the recirculation flow still keeps two distinct axes of rotations. The lip and the corner vortices are completely merged at $Wi = 5$. Table 6.3 reports the normalized length $X_R = x_R / h$ of the primary vortex at the reservoir corner, and the normalized intensities of the corner and the lip vortices, $\Psi_R = \left| (\phi_R / \phi_w) - 1 \right|$ and $\Psi_L = \left| (\phi_L / \phi_w) - 1 \right|$, respectively. The quantity x_R is the length of the corner vortex in contact with the horizontal walls of the reservoir, while ϕ_R and ϕ_L are the streamfunction values at the center of the corner and lip vortices, and ϕ_w is the boundary value of the streamfunction at the walls in contact with the vortices. The normalized values of the vortex length and the vortex intensities are plotted versus the Weissenberg number and compared to other numerical results from the literature, in Figure 6.21–6.23. The intensity of the lip vortex increases with the Weissenberg number, and eventually becomes larger than the intensity of the corner vortices, at $Wi = 1.5$.

After the transient development of the viscoelastic flows, the numerical solutions exhibit small fluctuations around a steady-state solution, for $Wi \geq 1.5$. The quantitative results provided hereinafter correspond to the averaged solution, after the transient development. Investigations of these fluctuations require numerical simulations for longer times, which were not in the scope of the present work.

Wi	X_R	$\Psi_R \times 10^3$	$\Psi_L \times 10^3$
0.0	1.50	1.17	0
0.5	1.46	1.03	0.20
1.0	1.39	0.85	0.72
1.5	1.32	0.67	0.75
2.0	1.24	0.53	1.1
2.5	1.17	0.43	1.5
3.0	1.12	0.39	2.0
3.5	1.11	0.44	3.0
4.0	1.12	3.7	
4.5	1.16	4.7	
5.0	1.26	6.5	
10	2.34	24.2	

Table 6.3: Normalized length of the primary vortex; normalized intensities of the corner vortex and the lip vortex, which start merging at $Wi = 3.5$.

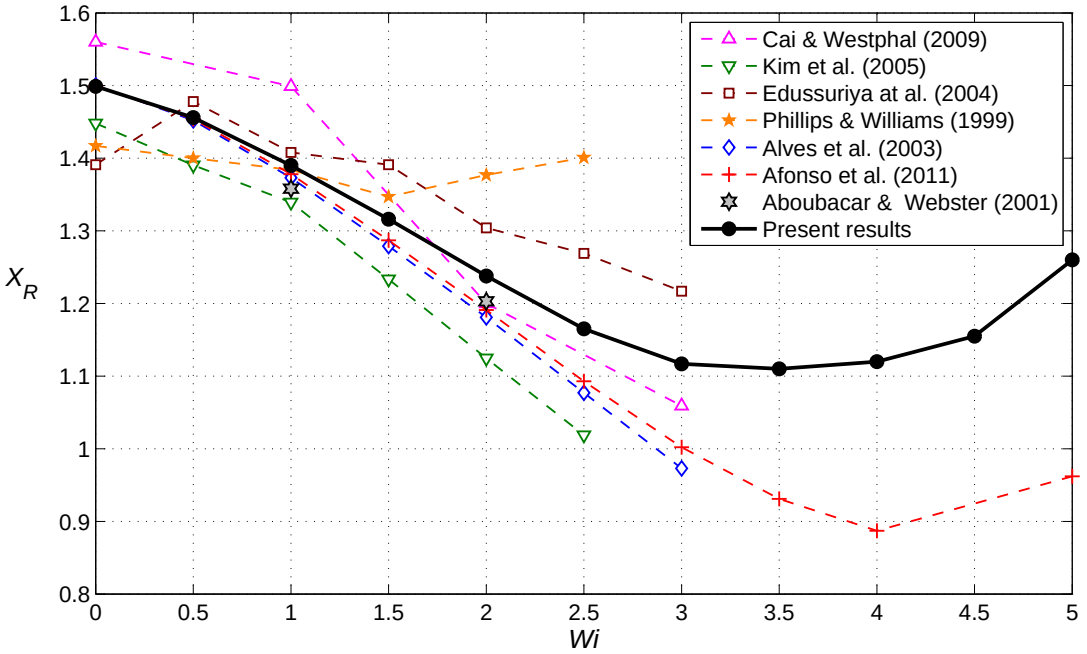


Figure 6.21: Normalized length of the corner vortex.

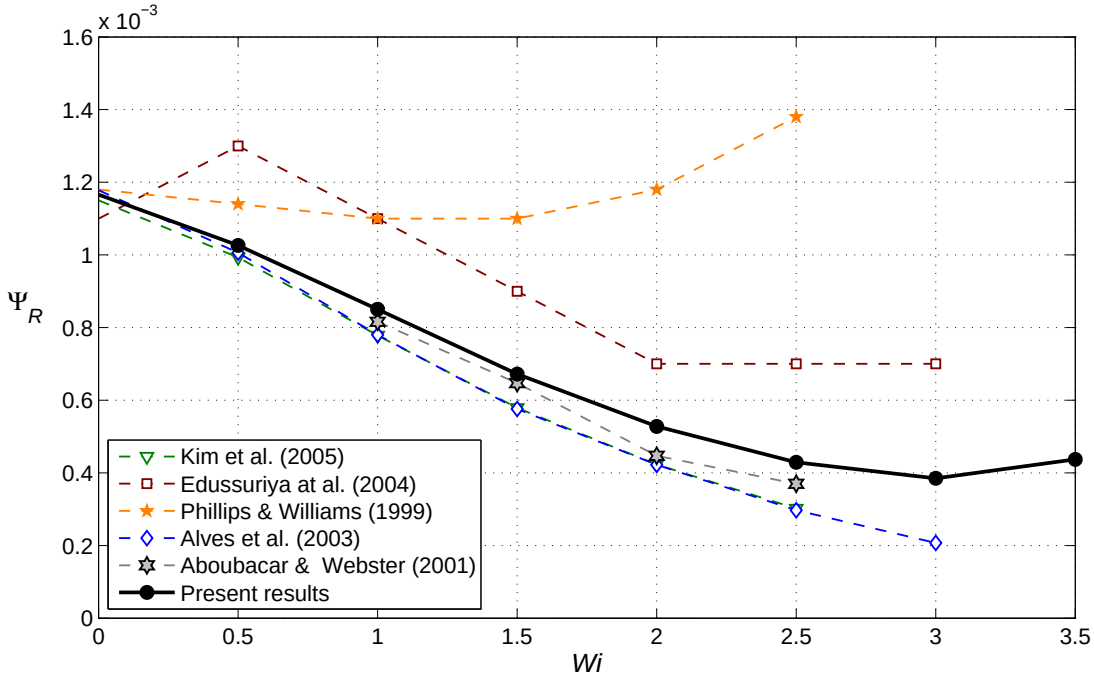


Figure 6.22: Normalized intensity of the corner vortex before the merging with the lip vortex.

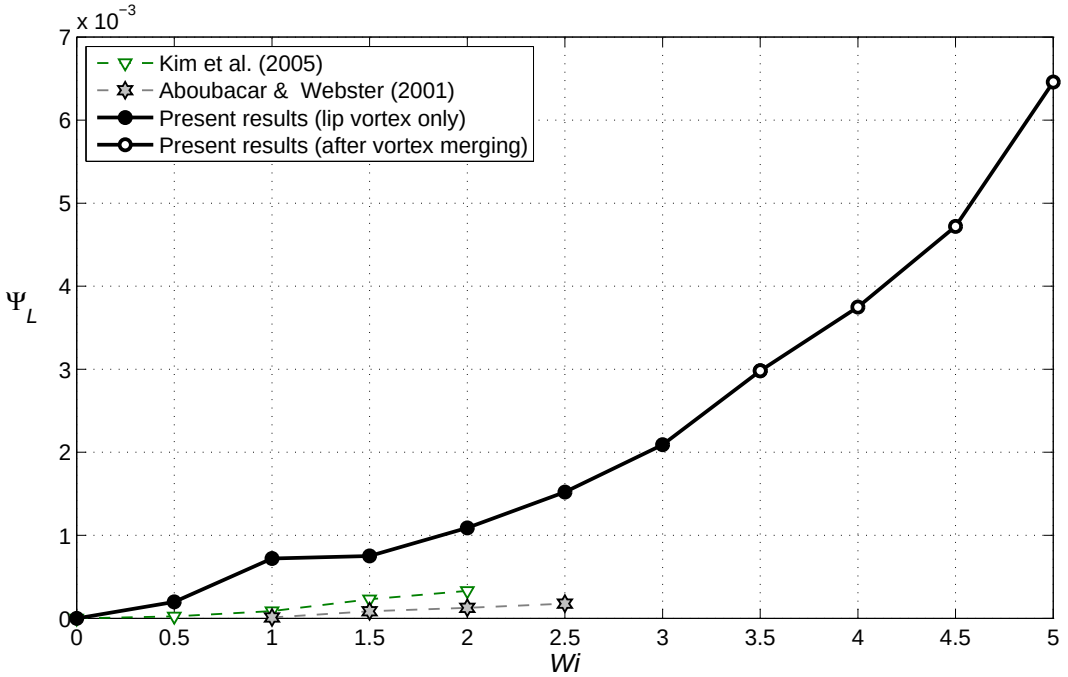


Figure 6.23 Normalized intensity of the lip vortex (before the merging) and the primary vortex (after merging).

The present results follow the same trend as the benchmark solutions of Alves, Oliveira and Pinho (2003b) and Afonso et al. (2011). However, there are discrepancies in the quantitative results of the different studies in the literature.

First of all, the accuracy of the solution depends on the discretization techniques used in the finite–volume or the finite–element algorithms, as shown by Alves et al. (2000, 2003a). Secondly, the vortex length and the vortex intensity are not the primary variables of the calculations based on the velocity–pressure formulation. Therefore, their precisions also depend on the post-processing calculations (the determination of the streamfunction from the velocity field requires the solving of a Poisson equation). In the present study, the streamfunction is the primary variable of the calculation, in contrast with the other numerical algorithms presented in the literature. The streamfunction values at the center of the vortices, which are local extrema, are obtained by biquadratic interpolation of the discrete (pointwise) streamfunction solutions. The calculation mesh in the vicinity of the reentrant corner is much finer in the present study than in the other simulations presented in the literature. This explains why the present solution enhances the values of the lip intensity.

The contours of the vortices, i.e. the isoline $\phi(\boldsymbol{x}) = \phi_w$, are determined by linear interpolation of the discrete streamfunction solutions. Then, the length x_R of the vortex in contact with the walls is obtained by extrapolation of the vortex contour to the wall. Hence, the final result for the vortex length also depends on the post-processing calculations. In the author's opinion, the reason why the present values of the vortex length are larger than those reported in the literature is because the particular choice of extrapolation procedure favors larger values. Therefore, the qualitative agreements with the benchmark solution of Alves, Oliveira, and Pinho (2003b) and Afonso et al. (2011) are satisfying, since the present post-processing calculations did not aim at reproducing the exact same values. Finally, the mesh density near the stagnation points of the corner vortices is relatively coarse, in the present simulations. More precise solutions could be obtained with further investigations using finer grids.

Numerical simulations have also been performed at the high Weissenberg numbers $Wi = 10$ and $Wi = 20$. At $Wi = 10$, the corner vortex grows in size and intensity as compared to the solution at $Wi = 5$, and the flow continues to experience small fluctuations. The transient solution also contains a *vortex back-shedding* regime in the outlet channel, where perturbations of the parallel flow originating from the outlet shed in the upstream direction, without back flow, until they annihilate, as shown in Figure 6.24. However, the present vortex back-shedding regime is believed to be a numerical artefact caused by the outlet boundary condition. Moreover, it does not impact the flow inside the abrupt contraction, since the outlet channel is very long. At $Wi = 20$, the viscoelastic flow exhibits elastic instabilities at the corner vortices, similar to the *third vortex growth* regime described by Afonso et al. (2011), see snapshots of the streamlines in Figure 6.25. However, only two cycles of the *third vortex growth* are present during the total simulation time $T_f = 200$; thus, we cannot draw a conclusion on the frequency (and a possible quasi-periodicity) of the *third vortex growth* instability. Figure 6.26 shows the 2-norm of the log–conformation tensor, before and after the elastic instability at the corner. As it can be seen, the elastic instability contributes to the dissipation of elastic energy accumulated near the contour of the corner vortices.

Again, more thorough investigations would be necessary to validate the present results, but the qualitative agreements with the simulations at high Weissenberg numbers of Afonso et al. (2011) are promising.

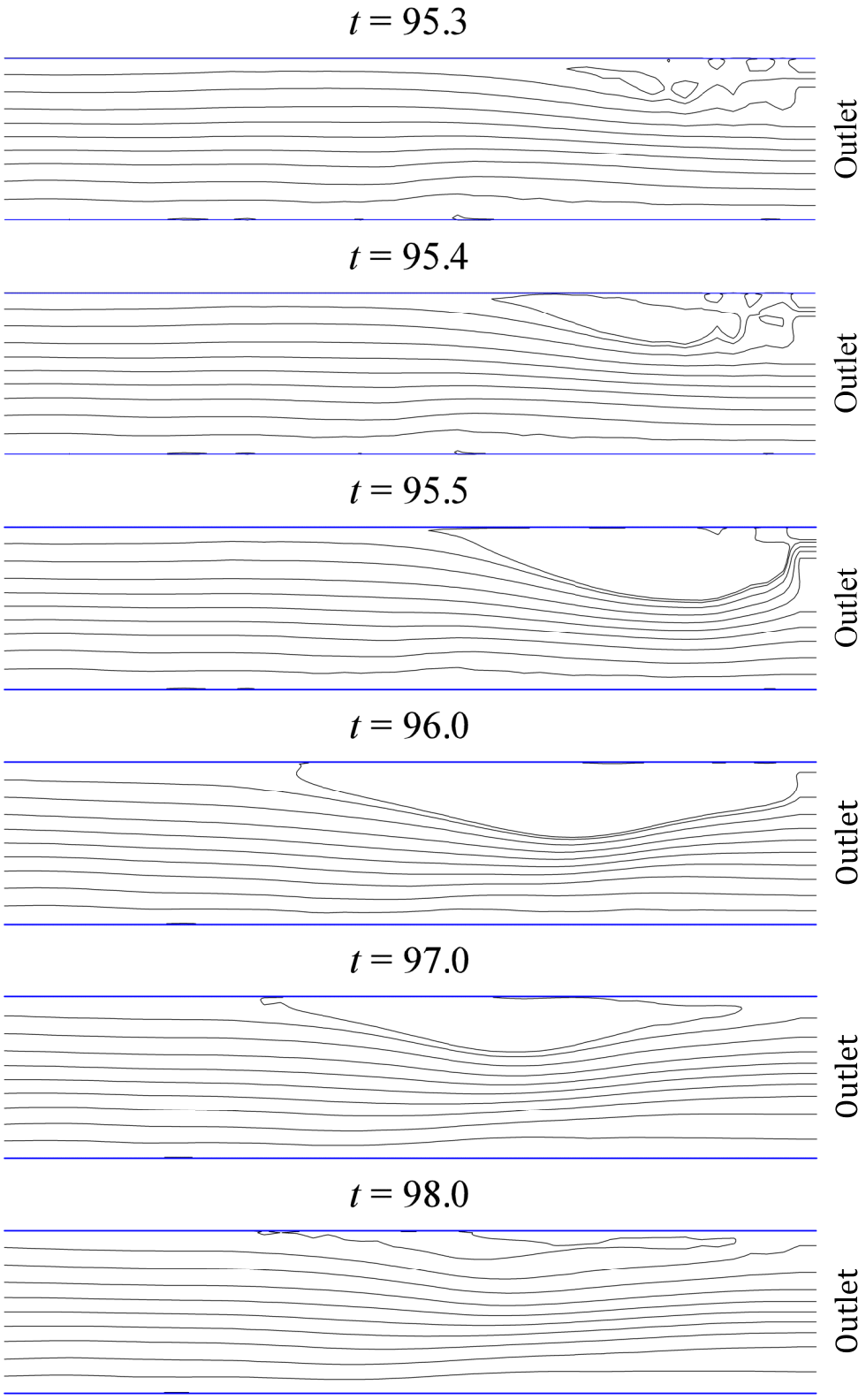


Figure 6.24: Streamlines of vortex back-shedding originating from the outlet, for $Wi = 10$.

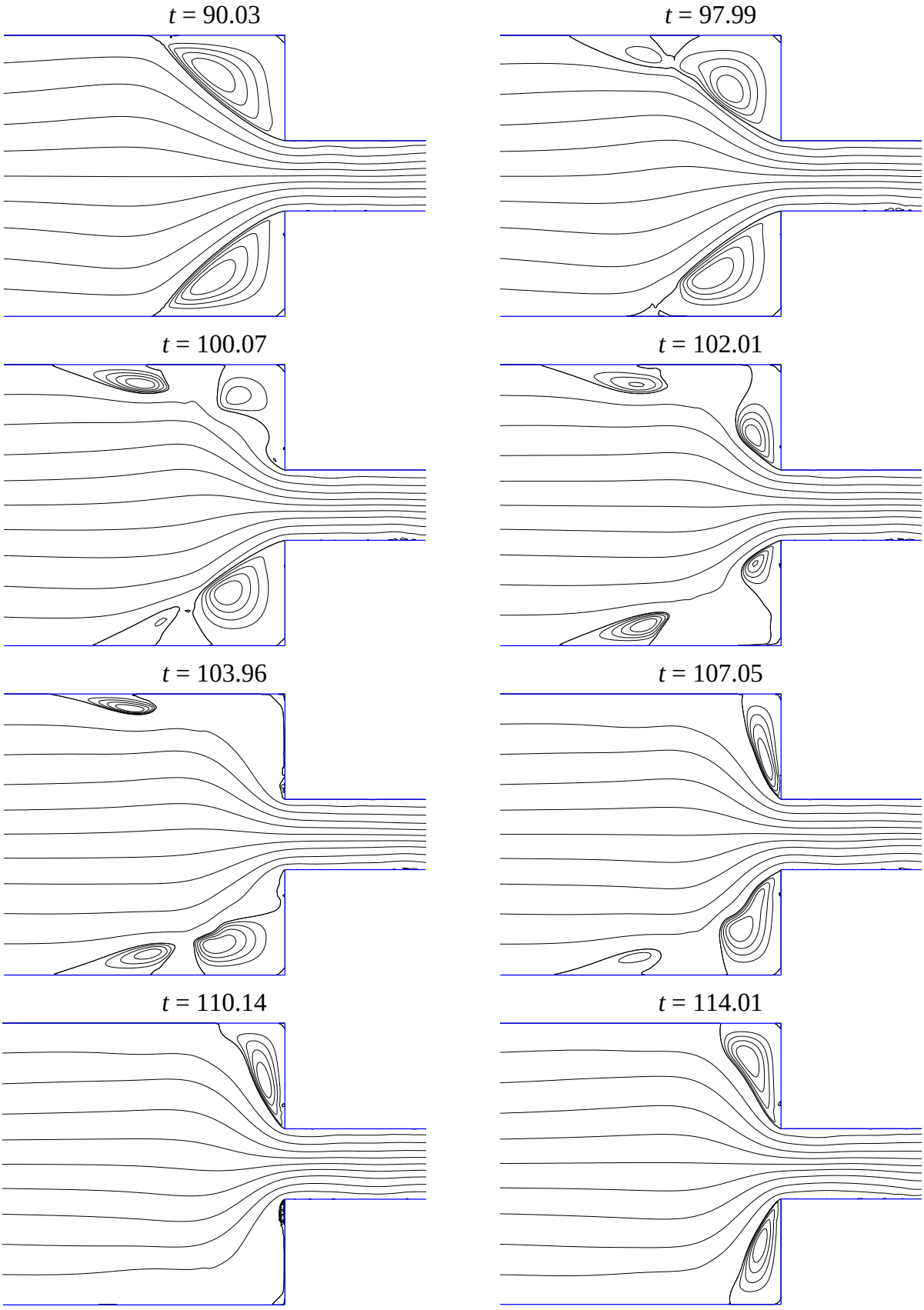


Figure 6.25: Streamlines of the elastic instability of the corner vortex in the 4:1 planar contraction, for $Wi = 20$.

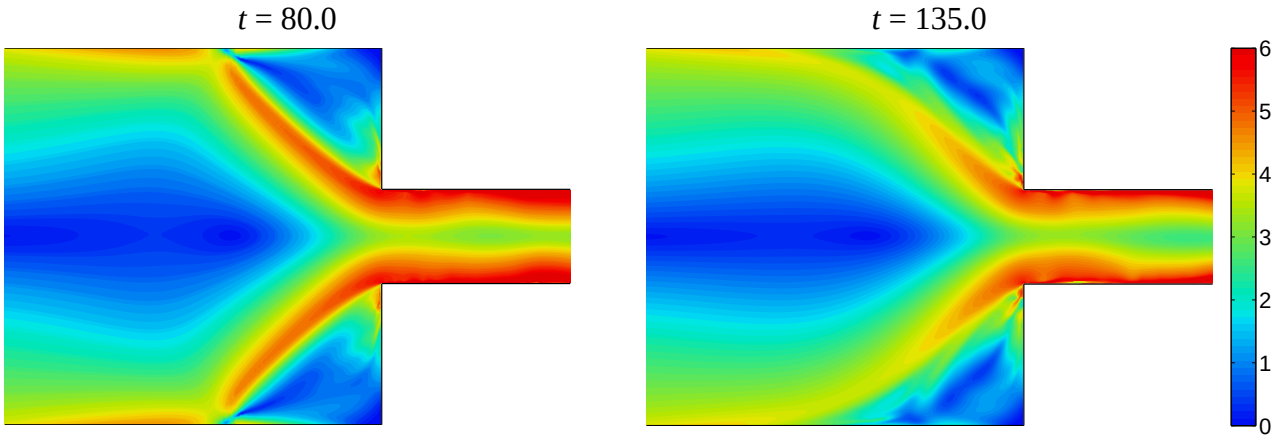


Figure 6.26: 2-norm of the log–conformation tensor, (left) before the elastic instability and (right) after the elastic instability, for $Wi = 20$. (Both pictures have the same color legend.)

6.3. Simulation of the extrudate swelling of an Oldroyd–B liquid

This subsection presents some preliminary results of the extrudate swelling of an Oldroyd–B liquid. The numerical simulation of extrudate swelling has a scientific interest as well as a practical importance in polymer processing. The planar extrudate swell has already been simulated by Tomé et al. (2002), who also use the Eulerian methodology, where the free-surfaces are tracked with the markers–and–cells algorithm. Numerical simulations of extrudate swelling within the Lagrangian framework have been done by Crochet and Keunings (1980; 1982a; 1982b), Russo and Phillips (2011), and Choi and Hulsen (2011), to cite a few. The simulations of viscoelastic free-surface flows of Bonito, Picasso and Laso (2006) with a variant of the Volume–of–fluid method, and Xu et al. (2012) with the smoothed particle hydrodynamics method, are also worth mentioning even though they solve flows in different geometries. In the present work, the free-surface of the extrudate is tracked with the *Volume–of–fluid* method and the *Cellwise Conservative Unsplit* advection scheme, presented in chapter 5.

The geometry of the simulations consists in a planar channel with an abrupt expansion. The parabolic fully-developed flow profile is imposed at the inlet of the channel. The external boundaries of the expansion are assigned the outlet boundary condition. The calculations take advantage of the symmetry, and only half of the geometry is simulated. The dimensions of the channel and the expansion are indicated in Figure 6.27. At the beginning of the simulation, half of the channel is already filled with the Oldroyd–B liquid. Note that the streamfunction flow solver also computes the flow of the air, which is assigned an artificial viscosity equal to a thousandth of the viscosity of the extruded material. The simulations do not include the gravity.

During the extrusion of a viscoelastic material, the first normal stress difference of the viscoelastic extra-stresses $N_1 = \tau_{\mathrm{P},xx} - \tau_{\mathrm{P},yy}$, which is developed inside the die, produces an additional swelling of the extrudate, as compared to the extrusion of inelastic liquids. The shear rate at the wall and the retardation ratio β are the key parameters that determine the level of the viscoelastic extra-stresses which are developed inside the channel. Thus, the Weissenberg number is here defined differently than in the previous sections. Here, the relaxation time is scaled by the wall's shear rate of the fully-developed flow profile:

$$Wi = \lambda \frac{2U_0}{h}, \tag{6.12}$$

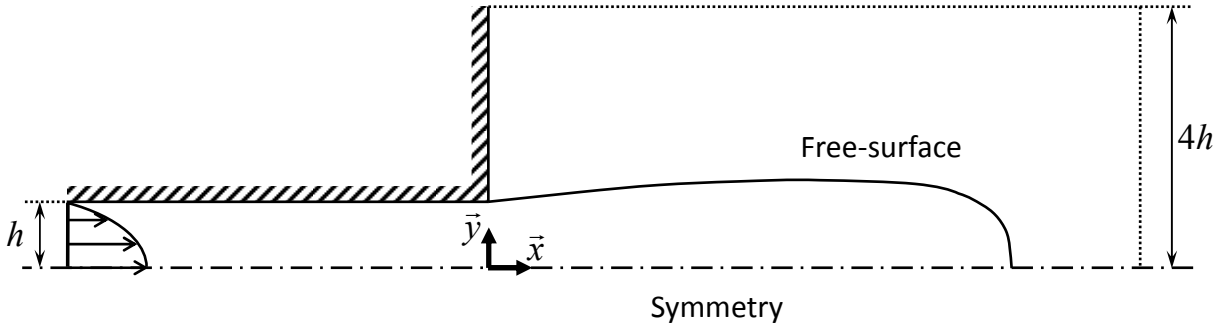


Figure 6.27: Geometry of the planar extrudate swelling simulations.

where U_0 is the maximum velocity of the fully-developed flow profile (i.e. the velocity at the symmetry plane), and h is half of the width of the channel. The retardation ratio of the Oldroyd–B fluid is taken as $\beta = 1/9$. The flows are simulated at the very low Reynolds number $Re = \rho U_0 h/\eta_0 = 8.3\times10^{-4}$.

Preliminary results show that the extrudate has a smooth surface, at low Weissenberg numbers. Figure 6.28 displays the different components of the velocity vector and the log–conformation tensor, for the value $Wi = 0.6$. It can be seen that the stress boundary layer and the velocity profile are relatively stable along the channel. However, the situation is quite different at higher Weissenberg numbers. Snapshots of the different components of the velocity vector and the log–conformation tensor for $Wi = 6$ are shown at different times in Figures 6.29–6.33. (Only half of the domain displayed in Figures 6.29–6.33 was actually computed; the other half is simply replicated by symmetry.) Large fluctuations in the stress layers and the velocity profile are noticeable. These fluctuations affect the flow at the die exit. The extrudate exhibits portions with different swelling ratios. The reader should note however that the simulated shape of the extrudate is constrained to the symmetry line. When the full geometry is simulated without the symmetry line, the fluctuations in the channel become asymmetric, and the extrudate alternately bends toward the top or the bottom of the vertical direction. This has a similarity with the fracture melt extrusion instability described in the introduction. However, the wavy and helicoidal instabilities observed in real extrusion are three-dimensional, whereas the two-dimensional bending in our simulations without a symmetry boundary is believed to be an artificial effect of the plane-strain assumption. Therefore, it was chosen to use the symmetry boundary condition. Although the symmetry clearly constraints the shape of the extrudate, we believe that it does not impact the mechanism of the instabilities that originate at the walls.

These extrusion instabilities seem to have a periodicity, but additional simulations are necessary to confirm this observation. At this point, we can say that the current simulations emulate the fracture melt extrusion defects, and that the observed instabilities are due to fluctuations in the stress layer at the wall. However, it cannot be concluded whether the stress instabilities have a physical origin or are due to a numerical artefact. Nevertheless, these observations are consistent with the theory of Morozov and van Saarloos (2007), which introduces the possibility of subcritical (nonlinear) flow instability in viscoelastic parallel flows, without the need of wall slip.

Finally, to the knowledge of the author, only one attempt of direct numerical simulation of the fracture melt extrusion defect has been reported in the literature to this day (Kwon 2015).

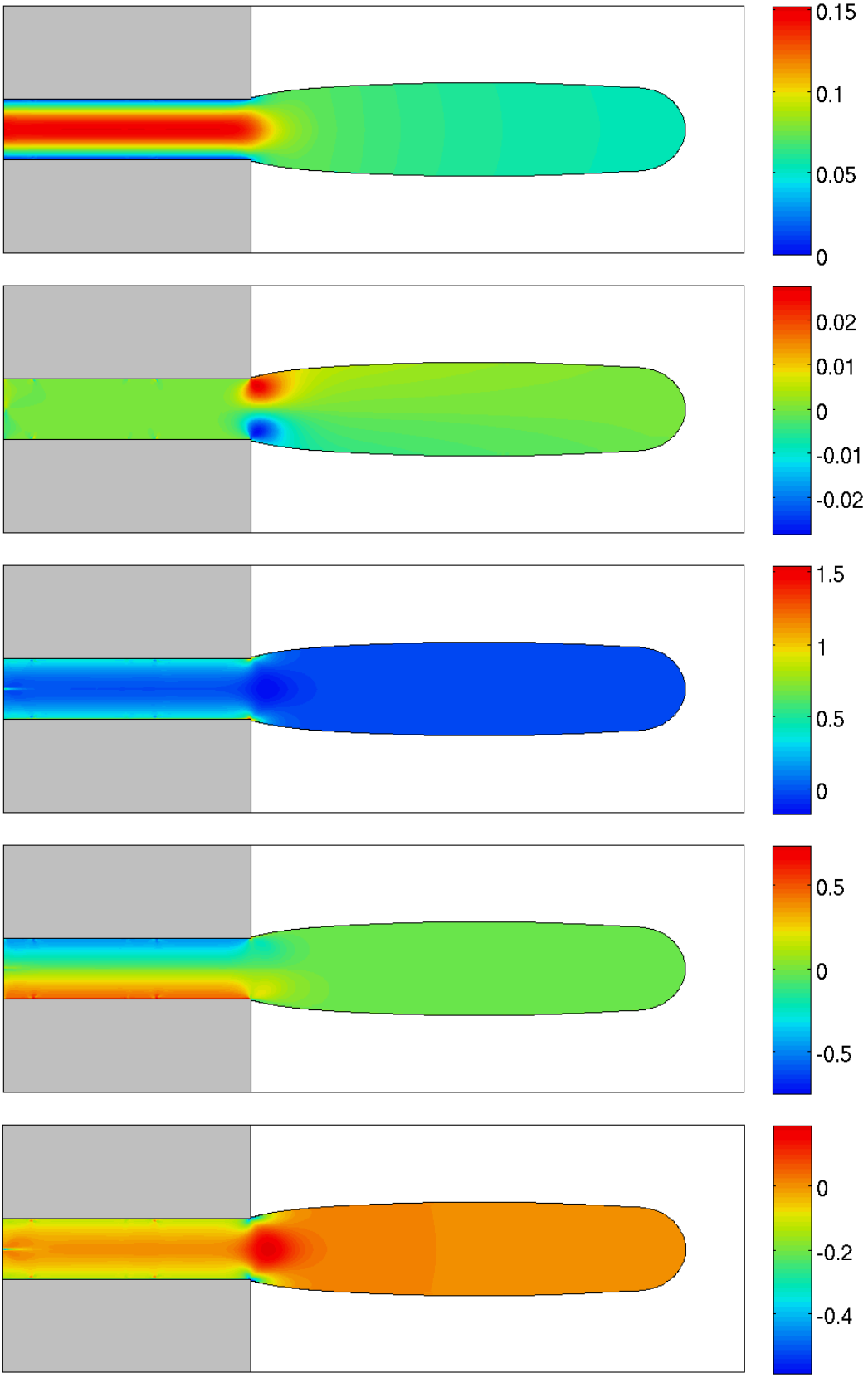


Figure 6.28: Stable extrusion at $Wi = 0.6$; from top to bottom: horizontal velocity, vertical velocity, Ψ_{11}, Ψ_{12} and Ψ_{22} component of the log–conformation tensor.

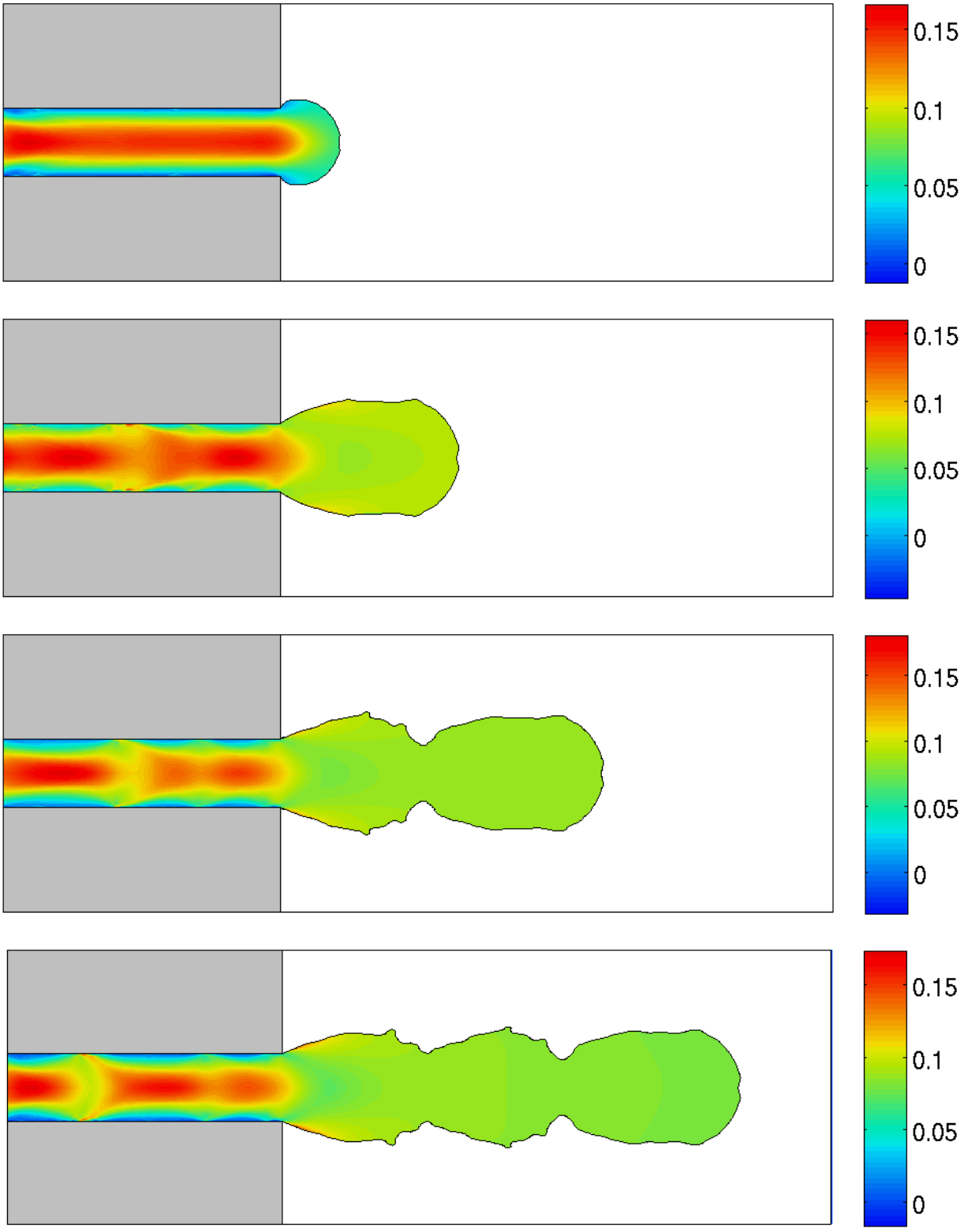


Figure 6.29: Snapshots of the horizontal velocity of the extrudate, at different times, for $Wi = 6$.

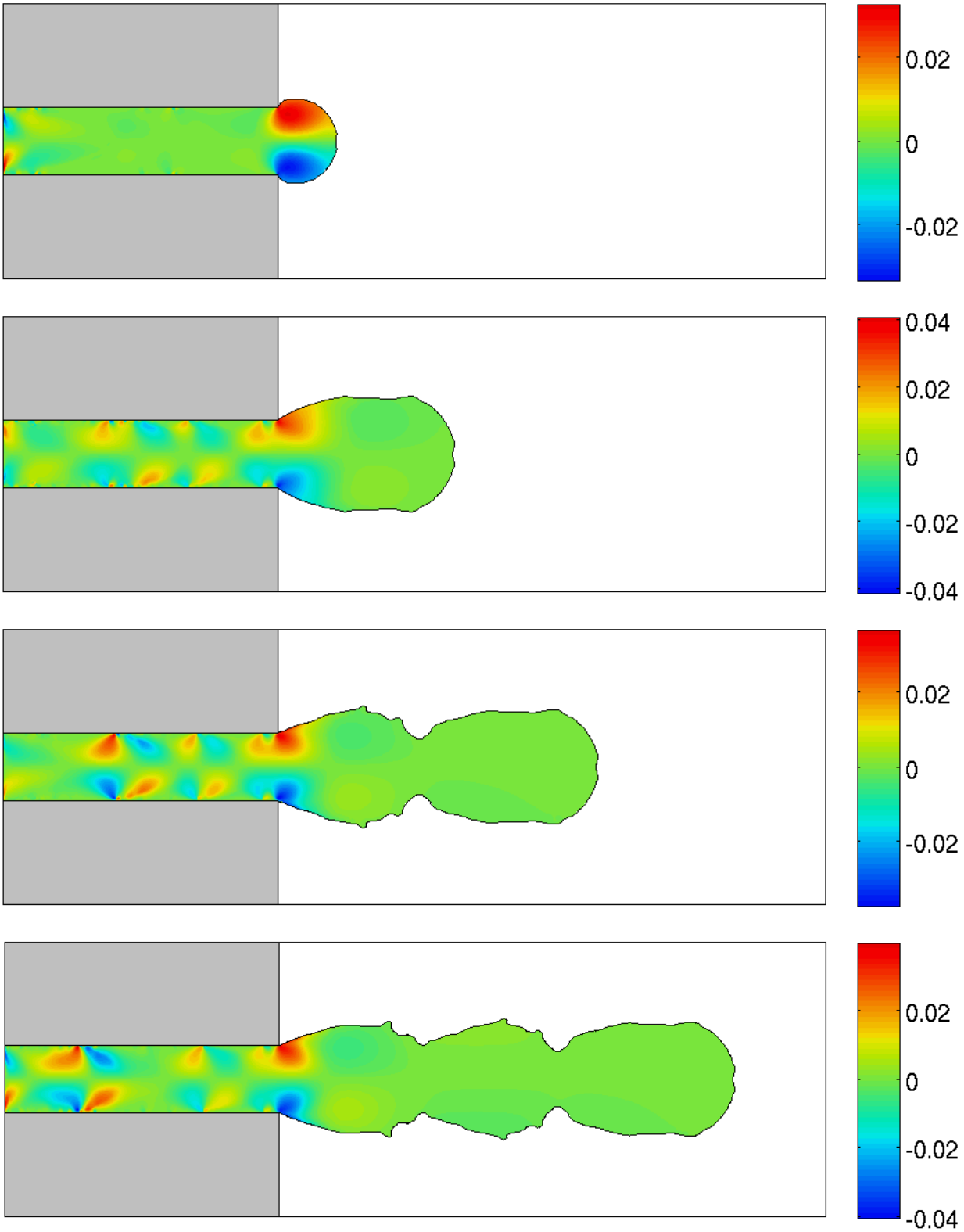


Figure 6.30: Snapshots of the vertical velocity of the extrudate, at different times, for $Wi = 6$.

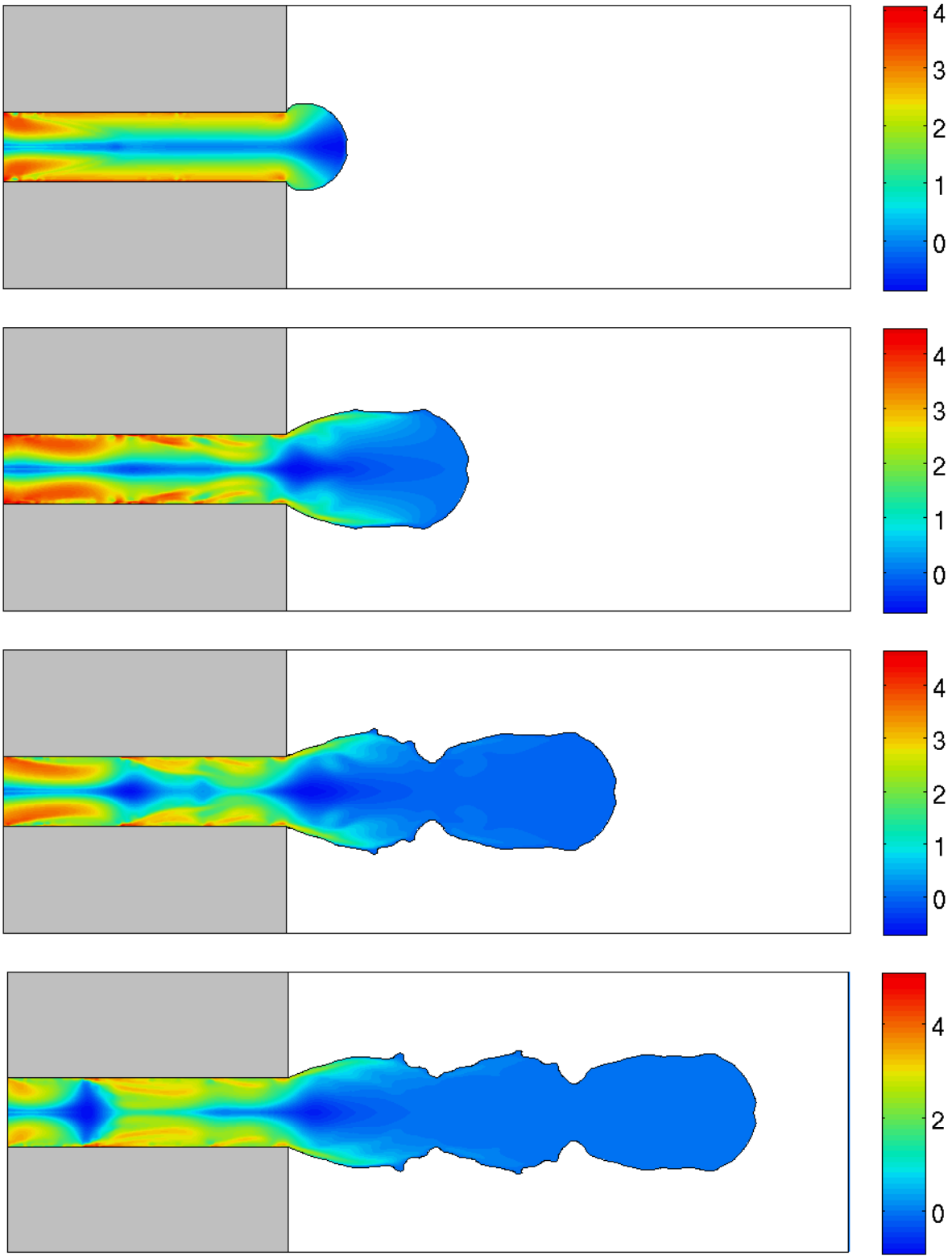


Figure 6.31: Snapshots of the component Ψ_{11} of the log–conformation tensor in the extrudate, at different times, for $Wi = 6$.

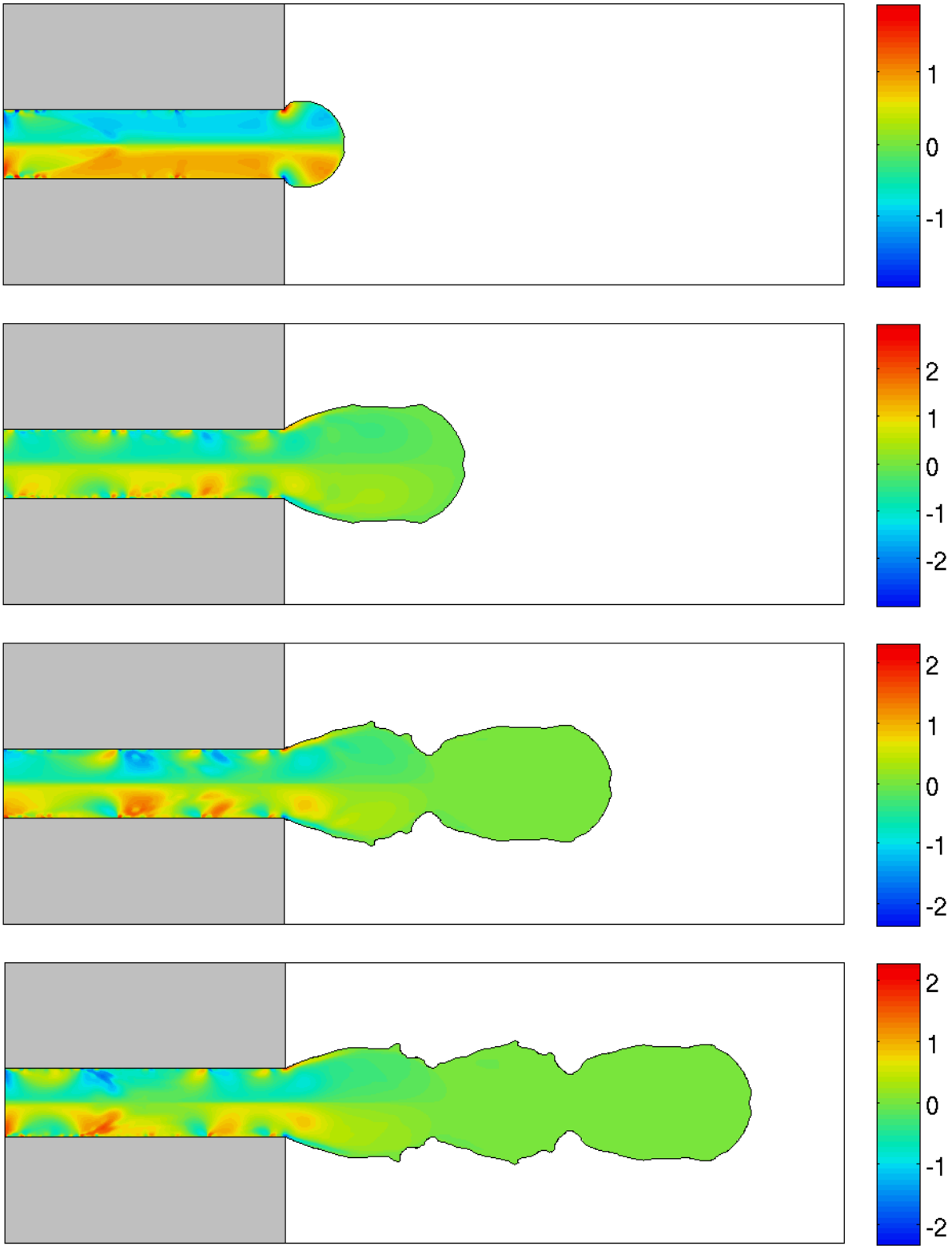


Figure 6.32: Snapshots of the component Ψ_{12} of the log–conformation tensor in the extrudate, at different times, for $Wi = 6$.

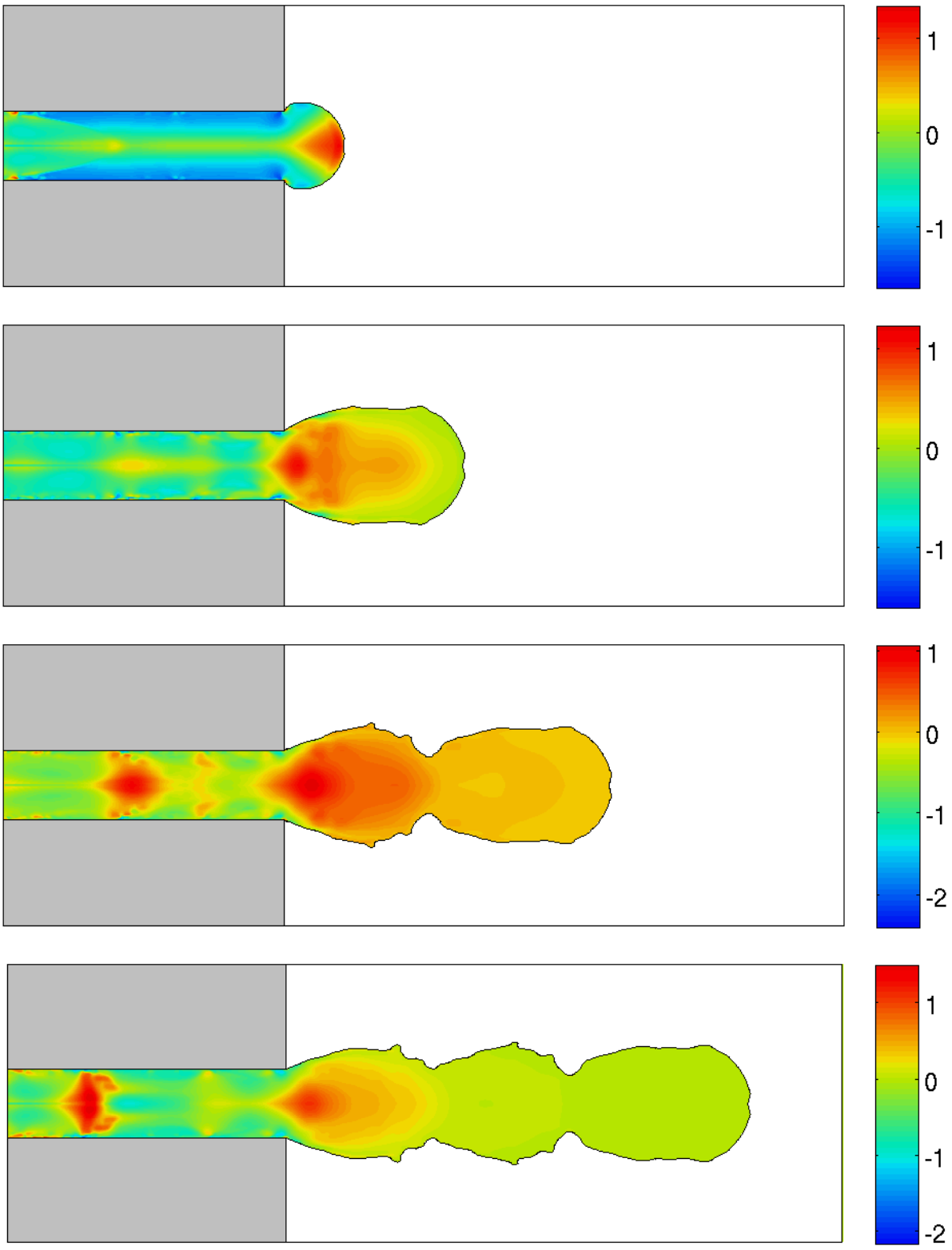


Figure 6.33: Snapshots of the component Ψ_{22} of the log–conformation tensor in the extrudate, at different times, for $Wi = 6$.

6.4. References

Aboubacar, M & Webster, MF 2001, 'A cell-vertex finite volume/element method on triangles for abrupt contraction viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 98, pp. 83–106.

Aboubacar, M & Webster, MF 2003, 'Development of an optimal hybrid finite volume/element method for viscoelastic flows', *International Journal for Numerical Methods in Fluids*, vol. 41, pp. 1147–1172.

Afonso, A, Oliveira, PJ, Pinho, FT & Alves, MA 2009, 'The log-conformation tensor approach in the finite-volume method framework', *Journal of Non-Newtonian Fluid Mechanics*, vol. 157, pp. 55–65.

Afonso, AM, Oliveira, PJ, Pinho, FT & Alves, MA 2011, 'Dynamics of high-Deborah-number entry flows: a numerical study', *Journal of Fluid Mechanics*, vol. 677, pp. 272–304.

Alves, MA, Oliveira, PJ & Pinho, FT 2003, 'A convergent and universally bounded interpolation scheme for the treatment of advection', *International Journal for Numerical Methods in Fluids*, vol. 41, pp. 47–75.

Alves, MA, Oliveira, PJ & Pinho, FT 2003, 'Benchmark solutions for the flow of Oldroyd-B and PTT fluids in planar contractions', *Journal of Non-Newtonian Fluid Mechanics*, vol. 110, pp. 45–75.

Alves, MA, Oliveira, PJ & Pinho, FT 2004, 'On the effect of contraction ratio in viscoelastic flow through abrupt contractions', *Journal of Non-Newtonian Fluid Mechanics*, vol. 122, pp. 117–130.

Alves, MA, Pinho, FT & Oliveira, PJ 2000, 'Effect of a high-resolution differencing scheme on finite-volume predictions of viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 93, pp. 287–314.

Belblidia, F, Matallah, H, Puangkird, B & Webster, MF 2007, 'Alternative subcell discretisations for viscoelastic flow: Stress interpolation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 146, pp. 59–78.

Belblidia, F, Matallah, H & Webster, MF 2008, 'Alternative subcell discretisations for viscoelastic flow: Velocity-gradient approximation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 151, pp. 69–88.

Boger, DV 1987, 'Viscoelastic flows through contractions', *Annual Review of Fluid Mechanics*, vol. 19, pp. 157–182.

Bonito, A, Picasso, M & Laso, M 2006, 'Numerical simulation of 3D viscoelastic flows with free surfaces', *Journal of Computational Physics*, vol. 215, pp. 691–716.

Brown, RA & McKinley, GH 1994, 'Report on the VIIIth international workshop on numerical methods in viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 52, pp. 407–413.

Cai, Z & Westphal, CR 2009, 'An adaptive mixed least-squares finite element method for viscoelastic fluids of Oldroyd type', *Journal of Non-Newtonian Fluid Mechanics*, vol. 159, pp. 72–80.

Choi, YJ & Hulsen, MA 2011, 'Simulation of extrudate swell using an extended finite element method', *Korea-Australia Rheology Journal*, vol. 23, pp. 147–154.

Crochet, MJ & Keunings, R 1980, 'Die swell of a Maxwell fluid: numerical prediction', *Journal of Non-Newtonian Fluid Mechanics*, vol. 7, pp. 199–212.

Crochet, MJ & Keunings, R 1982, 'Finite element analysis of die swell of a highly elastic fluid', *Journal of Non-Newtonian Fluid Mechanics*, vol. 10, pp. 339–356.

Crochet, MJ & Keunings, R 1982, 'On numerical die swell calculation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 10, pp. 85–94.

Edussuriya, SS, Williams, AJ & Bailey, C 2004, 'A cell-centred finite volume method for modelling viscoelastic flow', *Journal of Non-Newtonian Fluid Mechanics*, vol. 117, pp. 47-61.

Fattal, R & Kupferman, R 2004, 'Constitutive laws for the matrix-logarithm of the conformation tensor', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 281–285.

Fattal, R & Kupferman, R 2005, 'Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 126, pp. 23–37.

Hao, J & Pan, TW 2007, 'Simulation for high Weissenberg number: viscoelastic flow by a finite element method', *Applied Mathematics Letters*, vol. 20, pp. 988–993.

Kim, JM, Kim, C, Kim, JH, Chung, C, Ahn, KH & Lee, SJ 2005, 'High-resolution finite element simulation of 4:1 planar contraction flow of viscoelastic fluid', *Journal of Non-Newtonian Fluid Mechanics*, vol. 129, pp. 23–37.

Koplik, J & Banavar, JR 1995, 'Corner flow in the sliding plate problem', *Physics of Fluids*, vol. 7, pp. 3118–3125.

Kwon, Y 2004, 'Finite element analysis of planar 4:1 contraction flow with the tensor-logarithmic formulation of differential constitutive equations', *Korea-Australia Rheology Journal*, vol. 16, pp. 183–191.

Kwon, Y 2012, 'Numerical description of elastic flow instability and its dependence on liquid viscoelasticity in planar contraction', *Journal of Rheology*, vol. 56, pp. 1335–1362.

Kwon, Y 2015, 'Melt fracture modeled as 2D elastic flow instability', *Rheologica Acta*, vol. 54, pp. 445–453.

McKinley, GH, Raiford, WP, Brown, RA & Armstrong, RC 1991, 'Nonlinear dynamics of viscoelastic flow in axisymmetric abrupt contractions', *Journal of Fluid Mechanics*, vol. 223, pp. 411–456.

Morozov, AN & van Saarloos, W 2007, 'An introductory essay on subcritical instabilities and the transition to turbulence in visco-elastic parallel shear flows', *Physics Reports*, vol. 447, pp. 112–143.

Pakdel, P & McKinley, GH 1996, 'Elastic instability and curved streamlines', *Physical Review Letters*, vol. 77, pp. 2459–2462.

Pakdel, P & McKinley, GH 1998, 'Cavity flows of elastic liquids: purely elastic instabilities', *Physics of Fluids*, vol. 10, pp. 1058–1070.

Pan, TW, Hao, J & Glowinski, R 2009, 'On the simulation of a time-dependent cavity flow of an Oldroyd-B fluid', *International Journal for Numerical Methods in Fluids*, vol. 60, pp. 791–808.

Phillips, TN & Williams, AJ 1999, 'Viscoelastic flow through a planar contraction using a semi-Lagrangian finite volume method', *Journal of Non-Newtonian Fluid Mechanics*, vol. 87, pp. 215–246.

Phillips, TN & Williams, AJ 2002, 'Comparison of creeping and inertial flow of an Oldroyd B fluid through planar and axisymmetric contractions', *Journal of Non-Newtonian Fluid Mechanics*, vol. 108, pp. 25–47.

Renardy, M 2003, 'Stress integration for the constitutive law of the upper convected Maxwell fluid near the corners in a driven cavity', *Journal of Non-Newtonian Fluid Mechanics*, vol. 112, pp. 77–84.

Renardy, M 2006, 'A comment on smoothness of viscoelastic stresses', *Journal of Non-Newtonian Fluid Mechanics*, vol. 138, pp. 204–205.

Rodd, LE, Lee, D, Ahn, KH & Cooper-White, JJ 2010, 'The importance of downstream events in microfluidic viscoelastic entry flows: Consequences of increasing the constriction length', *Journal of Non-Newtonian Fluid Mechanics*, vol. 165, pp. 1189–1203.

Russo, G & Phillips, TN 2011, 'Spectral element predictions of die-swell for Oldroyd-B fluids', *Computers & Fluids*, vol. 43, pp. 107–118.

Sousa, PC, Coelho, PM, Oliveira, MSN & Alves, MA 2011, 'Effect of the contraction ratio upon viscoelastic fluid flow in three-dimensional square–square contractions', *Chemical Engineering Science*, vol. 66, pp. 998–1009.

Sousa, PC, Coelho, PM, Oliveira, MSN & Alves, MA 2009, 'Three-dimensional flow of Newtonian and Boger fluids in square–square contractions', *Journal of Non-Newtonian Fluid Mechanics*, vol. 160, pp. 122–139.

Thompson, RL & Mendes, PRS 2005, 'Persistence of straining and flow classification', *International Journal of Engineering Science*, vol. 43, pp. 79–105.

Tomé, MF, Mangiavacchi, N, Cuminato, JA, Castelo, A & McKee, S 2002, 'A finite difference technique for simulating unsteady viscoelastic free surface flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 106, pp. 61–106.

Xu, X, Ouyang, J, Jiang, T & Li, Q 2012, 'Numerical simulation of 3D-unsteady viscoelastic free surface flows by improved smoothed particle hydrodynamics method', *Journal of Non-Newtonian Fluid Mechanics*, vol. 177, pp. 109–120.

Xue, SC, Tanner, RI & Phan-Thien, N 2004, 'Numerical modelling of transient viscoelastic flows', *Journal of Non-Newtonian Fluid Mechanics*, vol. 123, pp. 33–58.

Yesilata, B, Öztekin, A & Neti, S 1999, 'Instabilities in viscoelastic flow through an axisymmetric sudden contraction', *Journal of Non-Newtonian Fluid Mechanics*, vol. 85, pp. 35–62.

Chapter 7

Conclusions

7.1. Summary

The simulation of viscoelastic flows has both a scientific importance and practical implications in polymer processing. This work has put the emphasis on the extrusion of polymeric materials, in particular. Despite the very simple geometry, the extrusion and co-extrusion processes are prone to dynamical instabilities because of the viscoelasticity of polymers. Chapter 1 has summarized the different mechanisms of the dynamical instabilities observed in extrusion and co-extrusion. The various modelling approaches that have been developed in the literature and the remaining modelling challenges have also been reviewed.

The popular differential constitutive models, derived from molecular theories of dilute polymer solutions, polymer networks, and entangled polymer melts of linear and branched polymers, have been surveyed in chapter 2. Inelastic phenomenological models describing shear–thinning and visco-plastic (yields stress) fluids, based on the generalized Newtonian fluid model, have also been provided. In addition, the numerical issues related to the *high Weissenberg number problem*, and its remedy with the log–conformation representation, have been discussed.

Chapter 3 deals with the formulations and the solution techniques of the conservation of mass and momentum. Drawbacks of the classical velocity–pressure decoupled method (which is by far the most popular approach) have been discussed. Finally, the pure–streamfunction formulation, which is derived from the vorticity–based methods, has been presented as an alternative to the pressure–based methods. The implicit pure–streamfunction formulation is formally more accurate than the velocity–pressure decoupled method, because it is immune of decoupling errors. Moreover, the absence of decoupling enhances the stability of the calculation.

The discretization of the governing equations (conservation laws and constitutive models), with the finite–volume method, on a Cartesian grid, has been described in chapter 4. Discrete curl operators are applied to the discretized momentum equations in order to obtain the matrix system of the discrete streamfunction variables. The coupling of the *streamfunction/log–conformation* scheme with adaptive under-relaxation and adaptive time-stepping yields an efficient viscoelastic flow solver algorithm. The potential extension of the method to three-dimensional simulations has also been discussed.

Chapter 5 addresses the modelling of multiphase flows with the Volume–of–Fluid (VOF) method, and the standard Piecewise–Linear–Interface–Construction (PLIC). In addition, a new *Cellwise Conservative Unsplit* (CCU) advection scheme has been developed. The CCU scheme updates the liquid volume fractions based on cellwise backward-tracking of the liquid volumes. The algorithm calculates non-overlapping and conforming adjacent

donating regions, which ensures the boundedness and conservativeness of the liquid volume. As a result, the CCU advection scheme is overall more accurate in classical benchmark tests, than the other state-of-the-art multi-dimensional VOF–advection schemes. In complex flows, the convergence rate of the CCU scheme with mesh refinements is between 2 and 3. Moreover, the remaining geometrical errors are mostly due to the inability of the standard PLIC to represent subgrid material topologies (i.e. high curvatures and thin material filaments), rather than the proposed CCU advection scheme.

The numerical method presented within this thesis has been implemented in Matlab. Examples of numerical results obtained with the proposed methodology have been reported in chapter 6. The performed simulations aimed at assessing the accuracy and the stability of the numerical method, rather than solving a specific problem. Thus, the Oldroyd–B constitutive model was chosen, despite its inadequate description of the viscoelastic extensional behavior, because it is more prone to numerical issues as compared to the more realistic nonlinear differential models. The calculations in the regularized lid–driven cavity exhibit second-order spatial and temporal accuracy. Moreover, simulations at very large Courant numbers are possible (up to 64 for low Weissenberg numbers), which demonstrates the robustness of the streamfunction/log-conformation scheme. Numerical simulations of the viscoelastic flows in the 4:1 planar contraction qualitatively agree with the benchmark solutions available in the literature. The enhancement of numerical stability, with the streamfunction/log–conformation scheme, makes it possible to simulation elastic instabilities at high Weissenberg numbers. The elastic instability appears to be a mechanism that dissipates the elastic energy stored inside the stress layers, in the regions with curved streamlines. Finally, preliminary simulations of extrudate swelling show that the fracture melt extrusion defect could be caused by instabilities in the stress layer at the surface of the die, triggered at moderate Weissenberg numbers. This instability could correspond to a non-linear subcritical viscoelastic instability of the parallel viscoelastic flows, but more work is necessary in order to conclude.

7.2. Perspectives

Further simulations of the *melt fracture* and the *stick-spurt* extrusion instabilities require the inclusion of realistic slip laws modelling *weak slip* and *strong slip* on the surface of the die. To the author's knowledge, simulation of extrusion that both includes the elastic instabilities originating from the reservoir and the complex slip behavior on the die surface, have never been done. Future work could also focus on the effects of the die geometry on these extrusion instabilities. In addition, the implementation of a thermal solver would make it possible to investigate the effects of viscous heating and temperature-dependent material properties.

Apart from physics that have been omitted, the present simulations are limited to relatively simple geometries, because of the Cartesian grid. The developed algorithm only allows rough representations of complex geometries, with stair-case contours. However, the pure–streamfunction formulation, the log–conformation representation and the VOF–PLIC–CCU methodology are not bounded themselves to Cartesian grids. Hence, the proposed approach could be extended to unstructured meshes, although it requires more involved programming. Then, the simulations of other benchmark flows, such as viscoelastic flows around a cylinder, in a corrugated channel or in a curved channel, could be addressed. Nevertheless, in the author's opinion, the use of unstructured meshes, which fit arbitrary geometries, is not necessarily the most suitable approach, although extensive research works have been done in the past decades toward the development of efficient mesh generators. The boundary fitted methods, where it is the geometries which are embedded into arbitrary grids, for instance with the *Cartesian cut–cell* approach or the *immersed–boundary–method*, are attractive alternatives to unstructured meshes. Moreover, a very high fidelity to the original geometries is achievable with the use of hierarchical quadtree grids. In addition, quadtree can easily be combined with

dynamical adaptive mesh refinements, which enhance the accuracy of the calculations in regions where it is needed.

The extension of the current algorithm to three–dimensional simulations is possible; however it will require the use of *high–performance–computing* technologies, to solve very large matrix systems. Indeed, the desirable three-dimensional solver must be able to deal with problems that have several millions of control–volumes (as modern commercial softwares do). For the purposes of comparison, the heaviest calculation performed with the developed in-house code (on a two–dimensional geometry) had about ~100,000 degrees of freedom. The computational cost for larger problems becomes too expensive for direct matrix solvers. Thus, it will be preferable to solve the large matrix systems of three-dimensional problems—which have more than ~100,000 unknowns—with the iterative solvers in the Krylov subspace family, as for instance the *Conjugated–Gradient* method, *BiCGSTAB*, *GMRES*, etc.

Cartesian grid systems also have the significant advantage of preserving the band-structure of the matrix system, which enhances the efficiency of the iterative solvers. In addition, the improvements of the discretization schemes toward high–order methods would be beneficial, in order to achieve more accurate calculations on a given mesh, or similar accuracy on coarser mesh (i.e. with fewer degrees of freedom). Concerning the extension of the CCU VOF–advection scheme to three-dimensions, the main challenge lies in performing complex geometrical operations, for instance computing the intersections of concave or self-intersecting polyhedra, with robust and efficient algorithms. Further research on the development of more accurate interface representations beyond the classical PLIC technique are also needed, in order to improve the accuracy of the VOF method.

Finally, the use of *Computational Fluid Dynamics* tools has become more and more popular in other domains of research beyond its traditional fields of applications. The simulation of multiphase viscoelastic flows has many other applications in different fields of Physics and Engineering, besides polymer processing, where it would be interesting to use the developed numerical framework. To cite a few examples, viscoelastic flows are very relevant to *Chemical Engineering*, *Micro-fluidics*, *Bio-mechanics* and *Environmental Science*.

Appendices

A. Paper–I

Comminal, R, Spangenberg, J & Hattel, JH 2013, 'Numerical simulations of viscoelastic flows with free surfaces', presented at *21ème Congrès Français de Mécanique*, Bordeaux, 26–30 August 2013.

Numerical simulations of viscoelastic flows with free surfaces

R. COMMINAL[a], J. SPANGENBERG[b], J.H. HATTEL[a]

a. *Technical University of Denmark (DTU), Produktionstorvet, 2800 KGS. LYNGBY (DENMARK)*
b. *Princeton University, PRINCETON, NJ 08540 (USA)*

Résumé :

Nous présentons une méthodologie pour simuler les écoulements viscoélastiques possédant des surfaces libres, dans la perspective de modéliser les procédés de mise en forme des polymères, tels que l'extrusion ou le moulage par injection. L'utilisation de lois de comportements viscoélastiques permet de prendre en compte les "effets de mémoire" des polymères déformés. Cependant, leurs utilisations engendrent aussi des difficultés d'ordre numériques, que nous résolvons avec une reformulation des contraintes viscoélastiques par la "log-conformation-transformation". Par ailleurs, la nouveauté de ces travaux réside dans l'utilisation de la méthode dite des "volume-of-fluid" pour simuler le déplacement des surfaces libres du fluide viscoélastique. Enfin, nous présentons des résultats de simulations préliminaires où nous testons séparément les différentes possibilités de notre modèle.

Abstract:

We present a new methodology to simulate viscoelastic flows with free-surfaces. These simulations are motivated by the modelling of polymers manufacturing techniques, such as extrusion and injection moulding. One of the consequences of viscoelasticity is that polymeric materials have a "memory" of their past deformations. This generates some numerical difficulties which are addressed with the log-conformation transformation. The main novelty of this work lies on the use of the volume-of-fluid method to track the free surfaces of the viscoelastic flows. We present some preliminary results of test case simulations where the different features of the model are tested independently.

Key words: computational rheology, viscoelastic flows, free-surface flows, volume-of-fluid method

1 Introduction

The simulation of viscoelastic flows with free surfaces is of particular interest for the modelling of polymer processing technologies, such as extrusion, injection moulding, blow moulding and tape casting, of single polymeric materials and ceramics colloidal suspension. Polymer solutions and melts are known for their non-Newtonian behaviours: *shear-thinning* (when the viscosity decreases with the deformation-rate), and *viscoelasticity* (when the deformations are affected by memory effects). Viscoelasticity is responsible for stress relaxation, creep deformations and elastic drawback. Those non-Newtonian behaviours fundamentally arise from the complex microstructure of the material, where long polymer chains are able to slide between each other, get stretched and change conformations during the deformations [1]. Non-Newtonian effects can also cause viscous and elastic interfacial instabilities in stratified flows [2]. This issue is important from both theoretical and practical points of views, especially for the co-processing of multi-material architectures.

Historically, the first studies of viscoelastic flows with free-surfaces were using Lagrangian methods (based on deforming meshes) in the context of finite-element analysis, to investigate the extrudate swell problem [3]. More recently, the *marker-and-cell* method [4] and the *level set* method [5] were implemented in order to track the free surfaces. Mixed Eulerian-Lagrangian methods were also developed [6]. In the present work, we use the *volume-of-fluid* (VOF) method, in which the free-surfaces are represented in a purely Eulerian manner. The main advantage of Eulerian methods over the Lagrangian methods is that they can adapt to arbitrary changes of topology of the free surfaces [7]. This is a very important feature for the simulation of injection moulding of complex geometries, in which junctions and separations of the material front frequently happen.

In the first part of the paper, we present the governing equations of the model. Then, the specific numerical techniques used to solve the viscoelastic flow problem and the free surfaces are briefly described. Each of those techniques has been implemented and tested separately with various test-case simulations. Finally, concluding remarks are summarized at the end of the paper.

2 Governing equations

The basic equations to be solved are the *continuity equation* (conservation of mass):

$$\nabla \cdot (\rho \boldsymbol{u}) = 0 \tag{1}$$

and the *momentum equations* (conservation of linear momentum):

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla p + \mu \nabla^2 \boldsymbol{u} + \nabla \cdot \boldsymbol{\sigma} \tag{2}$$

where ρ is the bulk density, μ is the (Newtonian) viscosity of the material, $\boldsymbol{u}$ is the velocity vector, p is the isostatic pressure and $\boldsymbol{\sigma}$ is the viscoelastic extra-stress tensor. In order to close this system of equations, a *constitutive model* which relates the viscoelastic stress tensor to the kinetics of the flow is required. The specificity of viscoelastic materials is that they behave both like liquids (viscous deformation) and solids (elastic deformation), depending on the time scale of observation or their rates of deformation. As a consequence, viscoelastic constitutive models cannot be expressed with algebraic relations, like for purely viscous and purely elastic materials. Indeed, viscoelastic constitutive models are either written on integral form or as partial differential equations (PDE), with respect to time. A large variety of models exist within those two classes of constitutive relations. On the one hand, some models were developed from an empirical approach, in order to fit particular experimental data best. On the other hand, generic models with the capability of describing most standard flow behaviours were derived, either from molecular kinetic theories or pure mathematical expansions [8]. In general, the appropriate constitutive model should be chosen considering its simplicity, its computational efficiency, its flexibility and its capability to describe the expected flow features. In this work, we use the *Oldroyd-B* model, which is one of the most popular viscoelastic constitutive relations. It is a linear partial differential constitutive model:

$$\lambda \frac{\mathfrak{D}\boldsymbol{\sigma}}{\mathfrak{D}t} + \boldsymbol{\sigma} = \eta \dot{\boldsymbol{\gamma}} \tag{3}$$

where λ is the relaxation time of the material, η is the viscosity of the viscoelastic deformations, $\dot{\boldsymbol{\gamma}} = \nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^{\mathrm{T}}$ is the strain-rate tensor, and

$$\frac{\mathfrak{D}\boldsymbol{\sigma}}{\mathfrak{D}t} = \frac{d\boldsymbol{\sigma}}{dt} + \boldsymbol{u} \cdot \nabla \boldsymbol{\sigma} - (\nabla \boldsymbol{u})^{\mathrm{T}} \cdot \boldsymbol{\sigma} - \boldsymbol{\sigma} \cdot (\nabla \boldsymbol{u}) \tag{4}$$

is the upper-convected time derivative of the stress tensor.

3 Numerical methods

3.1 Discretisation

The governing equations are discretized in space with the Eulerian finite-volume method. The calculation domain is divided in non-overlapping control volumes (CV), forming a structured orthogonal mesh of the geometry. The pressure and the viscoelastic stress component are stored at the centre of the CV, while the different velocity components stored at the centres of boundaries of the CV, as represented in figure 1(a). This staggered arrangement of the variables avoids the odd-even pressure decoupling. The equations of conservation (of mass and linear momentum) are integrated on each CV, and expressed in terms of a flux balance, via the divergence theorem. The evaluation of the fluxes at the boundary of the CV requires an interpolation. We use a quadratic upwind interpolation scheme with second-order accuracy in space. Moreover, the boundedness of interpolation is insured by the use of flux limiters, enhancing the stability of the scheme [9]. The resulting PDEs are further discretized in time with the two-step backward differentiation formula (BDF2), and the non-linear convective terms are linearized with a Taylor expansion around the solution at the previous time-step. It results in a stable implicit scheme with second-order accuracy in time.

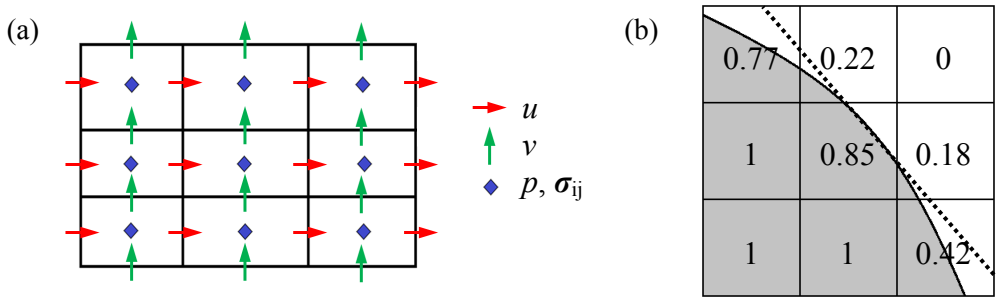


FIG. 1 – (a) Staggered variable arrangement on the mesh and (b) geometrical reconstruction of the free surface with a piecewise linear function.

The dimensionless quantity commonly used to measure the effect of viscoelasticity is the *Weissenberg number Wi*, defined as $Wi = \lambda\dot{\gamma}$, where $\dot{\gamma}$ is a characteristic deformation rate of the flow. When elastic effects are dominant (at high *Wi*), the solution of the constitutive equation is a fast exponential decay. It is a typical "stiff problem" prone to numerical instability. This difficulty is known as the « *High Weissenberg Number Problem* » [10]. It can be remedied by the passage to logarithmic variables in the constitutive equation [11,12]. It is done by taking the matrix-logarithm of the conformation tensor

$$\boldsymbol{c} = \boldsymbol{I} + \frac{\lambda}{\eta}\boldsymbol{\sigma} \tag{5}$$

because it is always *symmetric-positive-definite* (at the contrary to the viscoelastic stress tensor), which is a necessary condition for the existence of the matrix-log. The matrix-logarithm transformation also required the diagonalisation of the conformation tensor:

$$\log(\boldsymbol{c}) = \boldsymbol{R}\log(\boldsymbol{\Lambda})\boldsymbol{R}^{\mathrm{T}} \tag{6}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of the conformation tensor, and $\boldsymbol{R}$ is the orthogonal matrix of the eigendecomposition. This transformation yields a conservation equation for the log-conformation tensor $\boldsymbol{\Theta} = \log(\boldsymbol{c})$, which has a linear decay as solution. Without this transformation, the calculation of viscoelastic flows is still possible, but it is limited to *Wi* numbers below a threshold value depending on the kinetics of the flow and the discretisation techniques [13]. The results presented in the next section have been obtained without the use of the log-conformation transformation, in flow configurations below this limit.

3.2 Solution procedure

The velocities and the viscoelastic stresses can be calculated separately, by solving the momentum and constitutive equations sequentially. However, we do not have an evolution equation for the pressure, which is only seen in the momentum equations. The pressure is determined through the combination of the momentum and continuity equations instead. Indeed, the pressure can be seen as a Lagrange multiplier of the continuity equation, which limits the solution of the velocity to a divergence-free vector field. Therefore, the momentum and continuity equations have to be solved at the same time, but it is challenging from a numerical point of view, because the Jacobi matrix of the discretised system of equations is ill-conditioned. As a consequence, the numerical solutions are inaccurate and expensive to compute since only direct solvers, e.g. Gaussian elimination, can be used. The approach to remedy this problem is to derive a separate equation for the pressure, by taking the divergence of the momentum equations (the Chorin-Temam method). The resulting system of equations is further decoupled using a generalised LU-block decomposition [14]. We solve the problem efficiently with the GMRES iterative solver, based on conjugated-gradients.

3.3 Free surfaces tracking

Within the VOF method, a new discrete variable is introduced: the volume fraction $0 \leq \varphi \leq 1$ of the control volumes. At the end of each time step, the volume fraction is transported explicitly with the flow, and the apparent material properties of the CV are updated with the rule of mixture accordingly. The capillary forces

(surface tension) are neglected, as in this case they are several orders of magnitude lower than the viscous and elastic stresses. The advection scheme of the VOF method conserves the mass, and consists of two steps:

- The *geometrical reconstruction*: the interface inside cells where $0 < \varphi < 1$ is represented by a piecewise linear function. The position of the interface is parameterized by two variables: its slope and its distance from the centre of the control volume; and has one constraint: the volume fraction inside the cell. The remaining free parameter is determined so that the prolongation of the piecewise linear function into the eight neighbouring cells (in 2D) fits the best their volume fractions, see figure 1(b). This consists in an optimization problem. We use the ELVIRA method [15], which evaluates only six candidates for the interface, and chose the one which minimise the least-square error, without iterations. This method is of second-order accuracy.
- The *forward advection scheme*: the volume fraction of the fluid is advected explicitly, using the reconstructed position of the interface to evaluate the conservative fluxes. The volume fractions which are exchanged with the neighbouring cells are the one of the volumes effectively donated/accepted, rather than the total one of the cells. The time-step size of the calculation is constrained by the *Courant–Friedrichs–Lewy condition*, but the numerical diffusion of the scheme is limited. We used a split advection, and alternate the order of the directions of advection at every time-step. It was shown in [15] that in this way the advection is second-order accurate in time.

Unlike the front-tracking methods using markers or continuous chain of line segments to represent the interfaces, the VOF method naturally handles the junctions and separations of surfaces, without any need of surgical operations adding or removing boundary elements [7].

4 Results

This section presents the results of preliminary simulations performed in order to validate the implementation of our methods. The first test-case simulation is the transient viscoelastic Poiseuille flow driven by a pressure gradient suddenly imposed at the initial time. This simulation is used to validate the temporal accuracy of the algorithm, as it is one of the few transient problems for which an analytical solution exists. The exact solution does not have a closed form and is expressed as an infinite series, see [16]. The numerical simulation is performed at $Re = 0.02$, $Wi = 0.3$ and $\eta/\mu = 0.75$. Figure 2 presents the velocity magnitude at the central line of the channel, as well as the velocity profiles at different points of time. Unlike purely viscous fluids, here the velocity profile growths until it reaches a maximal amplitude, and then decrease slowly due to viscoelastic effects. The numerical results are in very good agreements with the analytical solution, and confirm the good temporal accuracy of the BDF2 scheme.

The second test-case simulation is the steady-state viscoelastic flow in planar contraction-expansion geometry. The flow inlet and outlet are the west and east boundaries of the domain respectively, where fixed pressures are imposed. The south boundary is a symmetry line, and all the other boundaries are solid walls with the no-slip condition. Flows in planar contraction have already been investigated, both theoretically [17,18] and numerically [19,20]. It was notably used to assess the spatial accuracy of different interpolation schemes [9,13]. Indeed, the different components of the velocity vector and the viscoelastic stress tensor are strongly coupled; the flow exhibits vortices at the salient corners, and stress singularities at the reentrent corners. However, the use of a staggered mesh arrangement avoids the calculation of the stress singularities. Figure 3 displays the contour plots of the pressure level and the viscoelastic stresses for a creeping flow ($Re = 0$) at $Wi = 1.04$ with $\eta/\mu = 1$. Unlike for purely viscous fluids, the viscoelastic creeping flow is unsymmetrical because of to the memory effects.

The last test-case simulation is the extrusion of a Newtonian fluid with free surfaces at $Re = 0$. The position of the free surface at the end of the extrusion is represented in the figure 4. The swelling ratio, defined as the ratio of the diameter of the extrudate by the diameter die, is $R_d/R_e = 1.45$. In this case, the swelling is due to dynamical effects only, as Newtonian fluids do not have memory effects. Indeed, the velocity profile, which has a parabolic shape at the die exit, has to rearrange itself to become a homogeneous velocity profile. We observe that it takes a distance of approximately $3R_d$ from the die exit for the material to obtain its final diameter. This simulation demonstrates the ability of the VOF method to track the free-surface in extrusion.

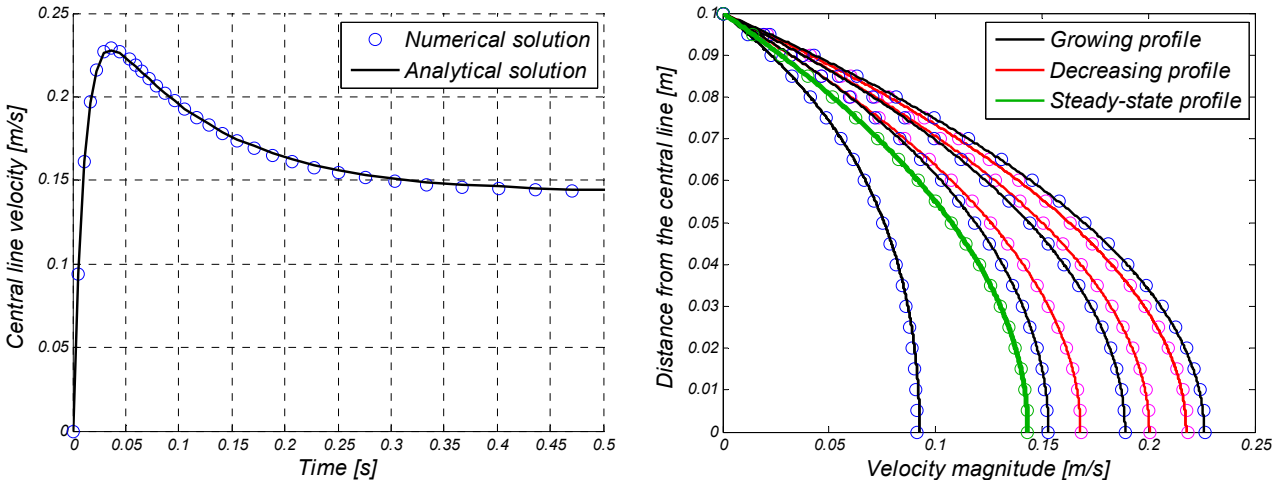


FIG. 2 – Transient viscoelastic Poiseille flow at $Re = 0.02$, $Wi = 0.3$ and $\eta/\mu = 0.75$. Left: velocity magnitude at the central line of the channel. Right: velocity profiles at the times [sec]: 0.005, 0.01, 0.015 (growing velocity profiles), 0.03, 0.09, 0.18 (decreasing velocity profiles), and 0.6 (steady-state solution).

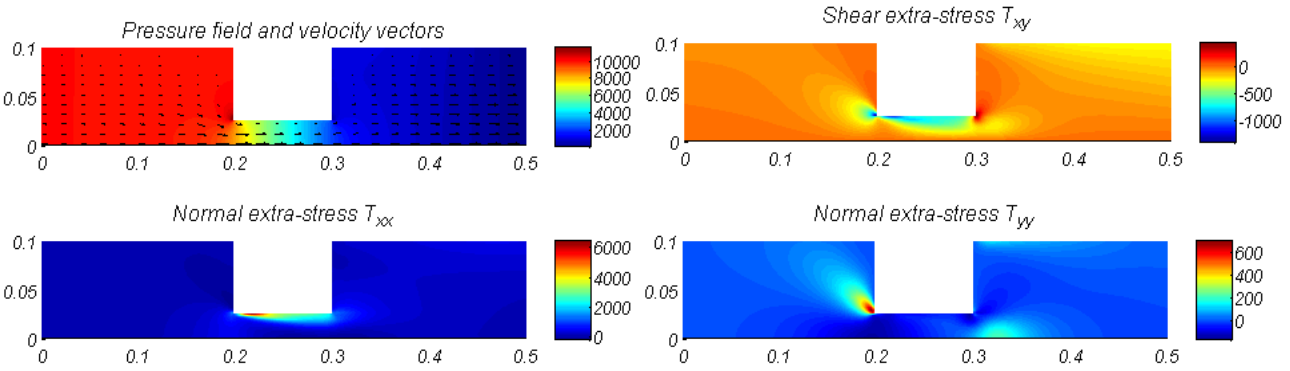


FIG. 3 – Contour plot of the pressure level and the viscoelastic stresses for the viscoelastic creeping flow in the planar contraction-expansion geometry at $Wi = 1.04$ with $\eta/\mu = 1$.

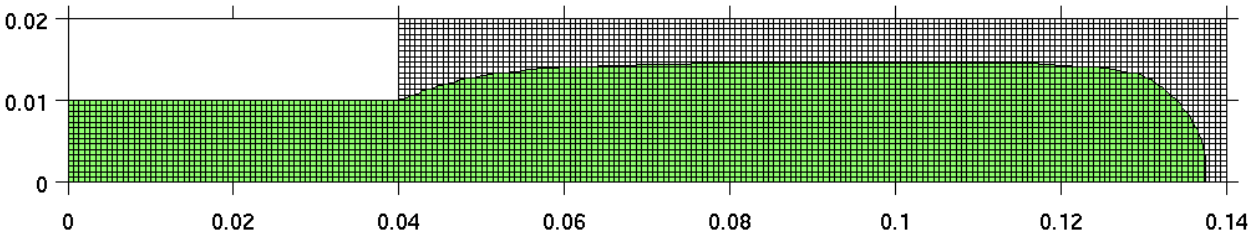


FIG. 4 – Position of the free surface at the end of the extrusion of a Newtonian material at $Re = 0$.

5 Concluding remarks

A numerical framework for the modelling of viscoelastic flows with free surfaces has been presented. The different features of the model: the viscoelastic stress solver and the free surface tracking algorithm, were tested separately. Their coupling has not been done yet, and it will be the focus of our future work. The preliminary results show good temporal and spatial resolutions, thanks to the second-order accuracy of the BDF2 and the upwind quadratic interpolation schemes. Moreover, the VOF method is a robust technique to track the free surfaces of viscoelastic flows, since it can adapt to changes of topology of the free surfaces. This is of great interest for the simulation of polymers manufacturing processes. This method can also be used to predict the interface deformations, due to the flow instability and the secondary recirculation flows generated by viscoelastic effects, in the co-processing of multi-materials architectures.

Acknowledgements

The authors would like to acknowledge the support of the Scientific Research Councils on Technology and Production Sciences (FTP) (Contract No. 09-072888, OPTIMAC), which is part of the Danish Council for Independent Research (DFF).

References

[1] Larson R.G., Constitutive Equations for Polymer Melts and Solutions, 1988.
[2] Dooley J., Hyun K.S., Hughes K., An experimental study on the effect of polymer viscoelasticity on layer rearrangement in coextruded structures, Polymer Engineering and Science, 38, 1060-1071, 1998.
[3] Crochet M.J., Keunings R., Finite element analysis of die swell of a highly elastic fluid, Journal of Non-Newtonian Fluid Mechanics, 10, 339-356, 1982.
[4] Tomé M.F., Mangiavacchi N., Cuminato J.A., Castelo A., McKee S., A finite difference technique for simulating unsteady viscoelastic free surface flows, Journal of Non-Newtonian Fluid Mechanics, 106, 61-106, 2002.
[5] Pillapakkam S.B., Singh P., A level-set method for computing solutions to viscoelastic two-phase flow, Journal of Computational Physics, 174, 552-578, 2001.
[6] Bonito A., Picasso M., Laso M., Numerical simulation of 3D viscoelastic flows with free surfaces, Journal of Computational Physics, 215, 691-716, 2006.
[7] Tryggvason G., Bunner B., Esmaeeli A., Juric D., Al-Rawahi N., Tauber W., Han J., Nas S., Jan Y.-J., A front-tracking method for the computations of multiphase flow, Journal of Computational Physics, 169, 708-759, 2001.
[8] Bird R.B., Wiest J.M., Constitutive equations for polymeric liquids, Annual Review of Fluid Mechanics, 27, 169-193, 1995.
[9] Alves M.A., Oliveira P.J., Pinho F.T., A convergent and universally bounded interpolation scheme for the treatment of advection, International Journal for Numerical Methods in Fluids, 41, 47-75, 2003.
[10] Keunings R., On the high Weissenberg number problem, Journal of Non-Newtonian Fluid Mechanics, 20, 209-226, 1986.
[11] Fattal R., Kupferman R., Constitutive laws for the matrix-logarithm of the conformation tensor, Journal of Non-Newtonian Fluid Mechanics, 123, 281-285, 2004.
[12] Fattal R., Kupferman R., Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation, Non-Newtonian Fluid Mechanics, 126, 23-37, 2005.
[13] Alves M.A., Pinho F.T, Oliveira P.J., Effect of a high-resolution differencing scheme on finite-volume predictions of viscoelastic flows, Journal of Non-Newtonian Fluid Mechanics, 93, 287-314, 2000.
[14] Perot J.B., An analysis of the fractional step method, Journal of Computational Physics, 108, 51-58, 1993.
[15] Pilliod J.E., Puckett E.G., Second-order accurate volume-of-fluid algorithms for tracking material interfaces, Journal of Computational Physics, 199, 465-502, 2004.
[16] Hayat T., Khan M., Ayub M., Exact solutions of flow problems of an Oldroyd-B fluid, Applied Mathematics and Computation, 151, 105-119, 2004.
[17] Hinch E.J., The flow of an Oldroyd-B fluid around a sharp corner, Journal of Non-Newtonian Fluid Mechanics, 50, 161-171, 1993.
[18] Davies A.R., Devlin J,, On corner flows of Oldroyd-B fluids, Journal of Non-Newtonian Fluid Mechanics, 50, 173-191, 1993.
[19] Aboubacar M., Matallah H., Webster M.F., Highly elastic solutions for Oldroyd-B and Phan-Thien/Tanner fluids with a finite volume/element method: planar contraction flows, Journal of Non-Newtonian Fluid Mechanics, 103, 65-103, 2002.
[20] Alves M.A., Oliveira P.J., Pinho F.T., Benchmark solutions for the flow of Oldroyd-B an PTT fluids in planar contractions, Journal of Non-Newtonian Fluid Mechanics, 110, 45-75, 2003.

B. Paper–II

Comminal, R, Spangenberg, J & Hattel, JH 2014, 'A two-phase flow solver for incompressible viscous fluids, using a pure streamfunction formulation and the volume of fluid technique', *Defect and Diffusion Forum*, vol. 348, pp. 9–19.

Defect and Diffusion Forum Vol. 348 (2014) pp 9-19
Online available since 2014/Jan/17 at www.scientific.net

doi:10.4028/www.scientific.net/DDF.348.9

A Two-Phase Flow Solver for Incompressible Viscous Fluids, Using a Pure Streamfunction Formulation and the Volume of Fluid Technique

Raphaël Comminal[1,a], Jon Spangenberg[2,b] and Jesper Henri Hattel[1,c]

[1]Department of Mechanical Engineering, Technical University of Denmark, Produktionstorvet, 2800 Kgs. Lyngby, Denmark

[2]Civil and Environmental Engineering, Princeton University, Princeton, NJ-08540, USA

[a]rcom@mek.dtu.dk, [b]js43@princeton.edu, [c]jhat@mek.dtu.dk

Keywords: Multiphase flow; streamfunction formulation; volume of fluid; Rayleigh-Taylor instability; broken dam problem

Abstract. Accurate multi-phase flow solvers at low Reynolds number are of particular interest for the simulation of interface instabilities in the co-processing of multilayered material. We present a two-phase flow solver for incompressible viscous fluids which uses the streamfunction as the primary variable of the flow. Contrary to fractional step methods, the streamfunction formulation eliminates the pressure unknowns, and automatically fulfills the incompressibility constraint by construction. As a result, the method circumvents the loss of temporal accuracy at low Reynolds numbers. The interface is tracked by the Volume-of-Fluid technique and the interaction with the streamfunction formulation is investigated by examining the Rayleigh-Taylor instability and broken dam problem. The results of the solver are in good agreement with previously published theoretical and experimental results of the first and latter mentioned problem, respectively.

Introduction

The simulations of multiphase and free-surface flows are of particular interest for the modeling of manufacturing processes involving molten materials, such as in metal casting or polymer molding. Free-surface flow simulations are useful to predict manufacturing defects related to improper mold filling. In addition, the development of advanced co-processing manufacturing techniques for multilayered composite structures and functionally graded materials (e.g. co-extrusion, co-injection molding, multilayer tape casting, etc.) requires the control of stratified flows. Depending on the flow conditions, the interface between immiscible fluids can be subjected to interfacial instabilities and deformations. In co-extrusion for instance, the mismatches of the rheological properties induce secondary flows which result in the deformation of the interfaces. Those phenomena are known as viscous encapsulation and elastic rearrangement [1]. In general, interfacial instabilities are generated when the two materials have different rheological properties, which lead to a discontinuity in the shear-rate across the interface. In this context, Computational-Fluid-Dynamics is a powerful tool to understand the kinetics of interfaces, and to improve the quality of production.

The main difficulty associated with incompressible flow is that the pressure field is not a state variable of the fluid. It is a dynamical parameter which does not have an evolution equation. In order to alleviate the difficulties with the velocity-pressure coupling, incompressible flows are generally solved with fractional-step methods, first introduced by Chorin [2]. It consists in splitting the Navier-Stokes problem into a general convection-diffusion equation for the momentum, and a Poisson equation for the pressure [3]. The semi-implicit methods (SIMPLER, SIMPLEC and PISO) can also be classified as fractional step method as they lay on the same idea of velocity-pressure decoupling [4]. Fractional step methods have demonstrated their efficiency to solve a wide range of aerodynamic and hydrodynamic problems, usually characterized by Reynolds numbers above unity. However, those methods present some disadvantages in the context of low-Reynolds number

flows, typical of polymer processing and microfluidic. Some of those disadvantages are listed below:

1) The decoupling between the velocity and pressure introduces splitting errors. Some methods can ensure a certain reduction rate of the splitting error when the time-step increment is reduced. However, an analysis of the splitting error with a Taylor series shows that the dominant error term is proportional to $(1/Re)^{\alpha}$, where α is a natural number [3]. As a consequence, the temporal accuracy of fractional step methods is severally degraded when $Re < 1$. Therefore, the convergence of the calculation might require a large number of iterations, and drastically small time-step increments.

2) The solution from the fractional-step methods does not fulfill the mass conservation to the machine precision [3]. Since the continuity equation is not solved directly, the accuracy of the mass conservation depends on the condition number of the Jacobi matrix for the velocity equation. It is a rule of thumb that the numerical results lose precision on a number of digits equal to the order of magnitude of the conditioning number. At low Reynolds number, the momentum equation is dominated by diffusive terms, and it results in a Jacobi matrix with very high condition number. Therefore, when $Re < 1$, the accuracy of the numerical results (and the mass conservation) is degraded [5].

3) The pressure equation requires boundary conditions which are not known a priori. There have been debates about the correct boundary condition to be used [6,7]. Moreover, its value at the boundary does not necessarily correspond to the physical pressure at the wall. It is rather a factitious value calculated so that the normal pressure gradient is compatible with the shear-rates at the wall. The uncertainty on the correct boundary condition can have dramatic effects, since the pressure equation is a boundary value problem (elliptic equation).

4) Errors in the pressure field have large impact on the motion of the interface. Indeed there is a coupling between the pressure field and the position of the interface. On one hand, multiphase flows at low Re numbers are driven by pressure gradients. (Viscous forces act as resistance forces). On the other hand the pressure field is directly related to the density field, which is determined by the interface position.

In this paper, we use an algorithm based on a pure streamfunction formulation, in order to avoid the disadvantages of the fractional step methods. The pure streamfunction formulation is more accurate than the fractional-step methods because it does not have any splitting errors. By construction it fulfills the mass conservation to the machine accuracy, without solving the continuity equation. In addition, it reduces the number of unknowns as it avoids the calculation of the pressure [8]. The pure streamfunction formulations have already been used for the simulation of single-phase flows [8,9]. The novelty of our method lies in the use of the streamfunction formulation for the simulation of multiphase flows of incompressible fluids. The Volume of Fluid (VOF) technique is used to track the position of the interface. The rest of this paper describes the derivation of the streamfunction evolution equation, and the numerical techniques used to solve it. We present results of 2D test case simulations that have been done to validate the implementation of our algorithm.

Governing Equations and Numerical Techniques

Derivation of the Streamfunction Formulation. The two-phase flow is modeled as a continuum single media, governed by the Navier-Stokes equations

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$

$$\frac{\partial(\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\nabla p + \nabla \cdot (\mu \dot{\boldsymbol{\varepsilon}}) + \rho \boldsymbol{g} \tag{2}$$

where $\boldsymbol{u}$ and the velocity vector, p is the isostatic pressure, ρ is the density, μ is the dynamic viscosity, g is the gravitational acceleration, and $\dot{\boldsymbol{\varepsilon}}$ is the deformation rate tensor defined as

$$\dot{\boldsymbol{\varepsilon}} = \frac{1}{2}\left[\nabla \boldsymbol{u} + \left(\nabla \boldsymbol{u}\right)^{\mathrm{T}} \right]. \tag{3}$$

The density varies in time and space when the interface moves, but the velocity field in the continuum media is divergence-free as both phases are incompressible. The density and dynamic viscosity depend on the phase of the flow, identified by the volume fraction θ. The convention we use is $\theta = 1$ inside the first phase and $\theta = 0$ inside the second phase. Then the apparent material properties of the media are obtained by the rule of mixture

$$\rho = \theta\rho_1 + \left(1-\theta\right)\rho_2 \tag{4}$$

$$\mu = \theta\mu_1 + \left(1-\theta\right)\mu_2 \tag{5}$$

where the subscript 1 refers to the material values of the first phase, and the subscript 2 refers to the material values of the second phase. The volume fraction θ is also transported with the flow, and it obeys an advection equation

$$\frac{\partial \theta}{\partial t} + \nabla \cdot \left(\boldsymbol{u}\theta\right) = 0. \tag{6}$$

The pressure-less formulation with streamfunction as primary variable replaces Eq. 1 and Eq. 2. It is derived in two steps. First, we take the curl of the momentum equation, in order to eliminate the pressure unknowns as $\nabla \times \nabla p = 0$. This provides an evolution equation for the vorticity

$$\boldsymbol{\omega} = \nabla \times \boldsymbol{u} \tag{7}$$

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{\omega} - \boldsymbol{\omega} \cdot \nabla \boldsymbol{u} = \nu \nabla^2 \boldsymbol{\omega} \tag{8}$$

where $\nu = \mu/\rho$ is the kinetic viscosity. Secondly, the divergence-free velocity field is expressed in term of a vector potential $\boldsymbol{\psi}$:

$$\boldsymbol{u} = \nabla \times \boldsymbol{\psi}. \tag{9}$$

The existence of the vector potential is proven by the Helmholtz decomposition theorem [10]. In two-dimension, only the out-of-plane component of the vector potential is of interest. It corresponds to the streamfunction ψ of the 2D flow. Finally, the evolution for the streamfunction in a 2D flow is:

$$\frac{\partial \nabla^2 \psi}{\partial t} + \frac{\partial \psi}{\partial y}\frac{\partial \nabla^2 \psi}{\partial x} - \frac{\partial \psi}{\partial x}\frac{\partial \nabla^2 \psi}{\partial y} = \nu \nabla^4 \psi \tag{10}$$

where ∇^2 is the Laplace operator and ∇^4 is the biharmonic operator. There is no need to solve the continuity equation anymore, because the incompressible constraint is automatically fulfilled by the streamfunction as $\nabla \cdot (\nabla \times \psi) = 0$.

Numerical Techniques. The flow problem is address within an Eulerian framework, i.e. the material flows through a fixed mesh which is in general not aligned with the interface. The streamfunction evolution equation Eq. 10 contains fourth-order derivatives. Its discretization with

finite differences would require boundary conditions for the first, second and third derivatives of the streamfunction. In order to avoid this difficulty, we first build up a matrix system for the momentum equations, containing the physical boundary conditions for the velocities. Then the discretized momentum equations are projected into the streamfunction space. The information of the streamfunction derivatives is contained in the boundary conditions for the tangential velocities. The momentum equations are discretized using the finite volume method, on a structured mesh, with a staggered arrangement of the velocities unknown. We use the two-level backward differentiation formula for the time integration, and a quadratic upwind interpolation scheme with flux limiters [11] for the interpolation of the convective flux at the faces of the control volumes. For the projection of the equation, the Jacobi matrix of the momentum equations is multiplied by the discrete curl operators of the velocity and the streamfunction, in the same way as in [8]. Finally the matrix system for the streamfunction only requires Dirichlet boundary conditions, which are obtained from the boundary conditions of the normal velocities. Finally, the resulting matrix system is solved by a direct solver.

The transport equation Eq. 6 is solved with the VOF method, which produces much less numerical diffusion than if it was solved with an ordinary finite-volume or finite element method. The ELVIRA technique is used to approximate the interface as piecewise linear functions inside each control volume, and the volume fraction is advected explicitly with a split donor-acceptor scheme [12]. Fixed point iterations are used between the streamfunction solver and the VOF algorithm in order to improve the solution at the end of the time step. The iterations are stopped when the Euclidean norm (2-norm) of the solution changes reaches a value less than 10^{-6}. Unusually, the convergence criterion is reached after a small number of iterations.

Numerical Examples

Rayleigh-Taylor Instability. The Rayleigh-Taylor instability is an interfacial instability driven by the gravitational force, when a heavy fluid is located on top of a lighter one. Rayleigh-Taylor instabilities have been extensively studied, and were used as a test case problem to validate multiphase algorithm [13-15]. Here we address the Rayleigh-Taylor instability of viscous fluids. The simulation setup is a rectangular box of width $w = 0.01$ and height $h = 0.04$, where the interface between the heavy fluid (at the top) and the light fluid (at the bottom) is initially located at the horizontal midline. This configuration is an instable equilibrium that breakdowns when a small perturbation is added to the interface. The no-slip boundary condition is applied at the top and bottom of the box. We apply the slip boundary condition at the left and right side of the box in order to emulate a repetition of the perturbation. The calculation domain is discretized with 50×200 uniform grid.

Linear stability analyses, assuming a potential flow, predict that the amplitude of the perturbation growths exponentially in time, with a growth rate n given by [16]:

$$n = \sqrt{gkA} \tag{11}$$

where g is the gravitational acceleration, k is the wavenumber of the perturbation, and $A = (\rho_1 - \rho_2)/(\rho_1 + \rho_2)$ is the Atwood number which quantifies the density ratio of the two fluids. However this theoretical growth rate is only valid in the range of small-amplitude perturbations. In our simulation, the initial interface is perturbed at $t = 0$ with a small sinusoidal wave of amplitude 0.001 and wavelength w. The gravitational acceleration is set to $g = -10$. We investigate two cases, when $\rho_1 = 1$, $\rho_2 = 0.1$, and when $\rho_1 = 1$, $\rho_2 = 0.6$. Those two cases give Atwood numbers of 0.818 and 0.25, respectively. The fluids have the same dynamical viscosity of $\mu = 5 \cdot 10^{-5}$ in both simulation. Snapshots of the interface profiles at different times are represented in Fig. 1 and Fig. 2, for $A = 0.818$ and $A = 0.25$, respectively.

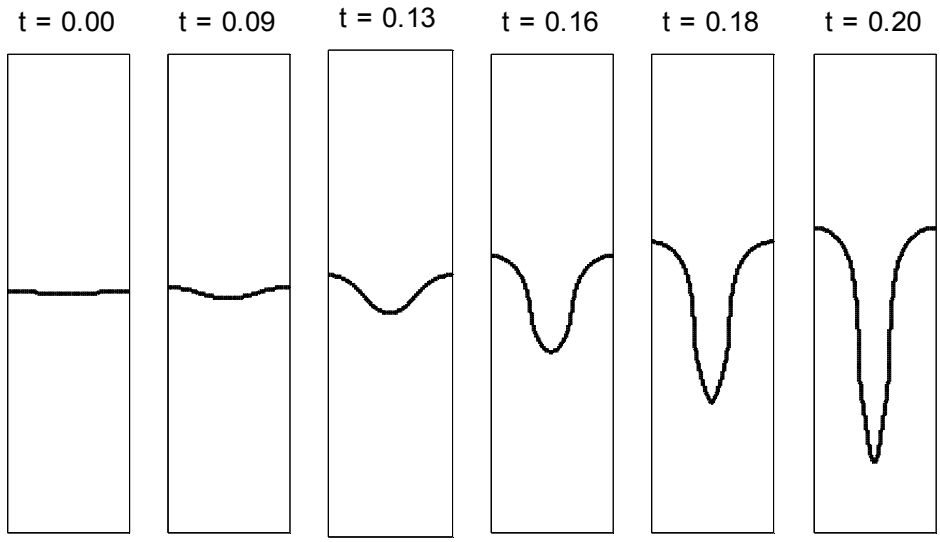


Figure 1. Evolution of the perturbed interface into a spike for $A = 0.818$.

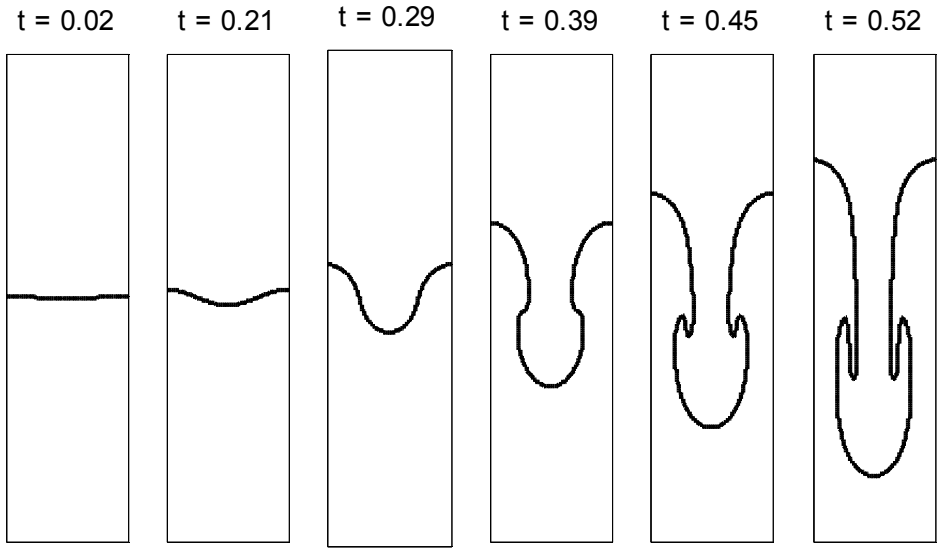


Figure 2. Evolution of the perturbed interface into a mushroom cap for $A = 0.25$.

The amplitude of the perturbation is plotted versus time, in Fig. 3 and Fig. 4. In the regime of small-amplitude deformation, the exponential growth rate fits very well the theoretical prediction. For $A = 0.25$ we observe a short delay of the perturbation growth compared to the linear stability theory, which is due to the relative effect of viscosity acting against the interface instability. In the later stage of the simulation, the growth rate deviates from the linear stability theory and tends to a linear growth regime. At high Atwood number, the interface deform as a spike, while at low Atwood number we observe the formation of a mushroom cap. The vorticity field and the streamfunction of both structures are displayed in Fig. 5, to help the visualization of the dynamics. Our results agreement qualitatively well with other calculations [13,15,17,18].

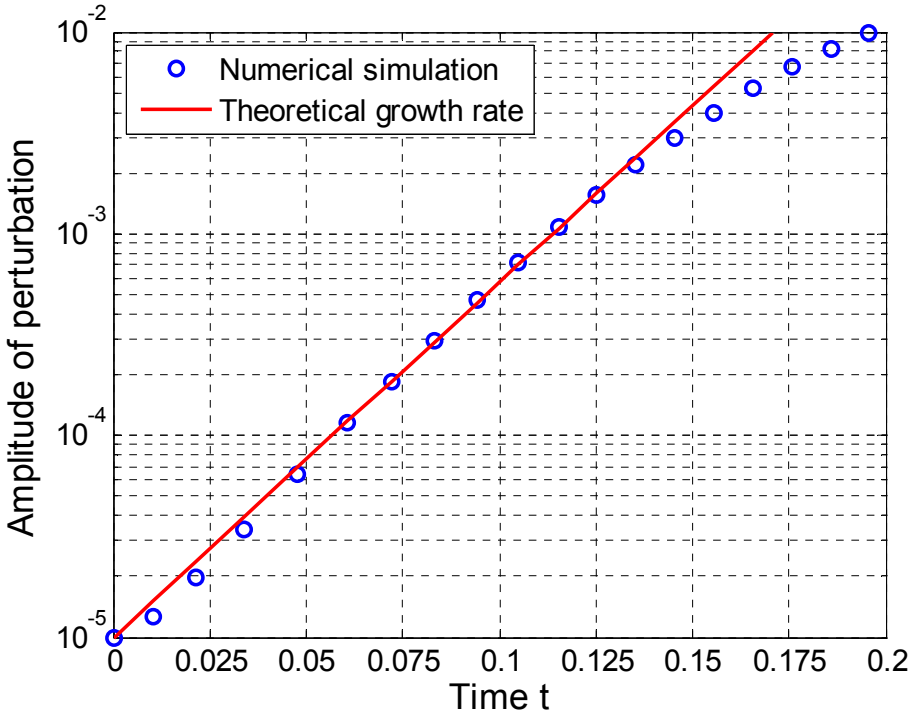


Figure 3. Growth of the small-amplitude perturbation for $A = 0.818$.

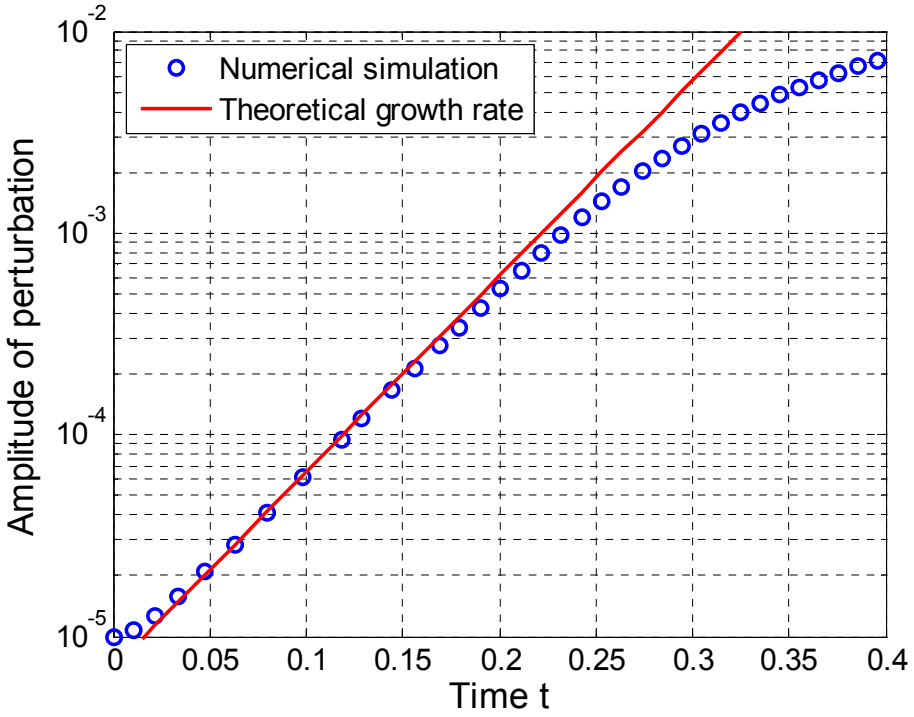


Figure 4. Growth of the small-amplitude perturbation for $A = 0.25$.

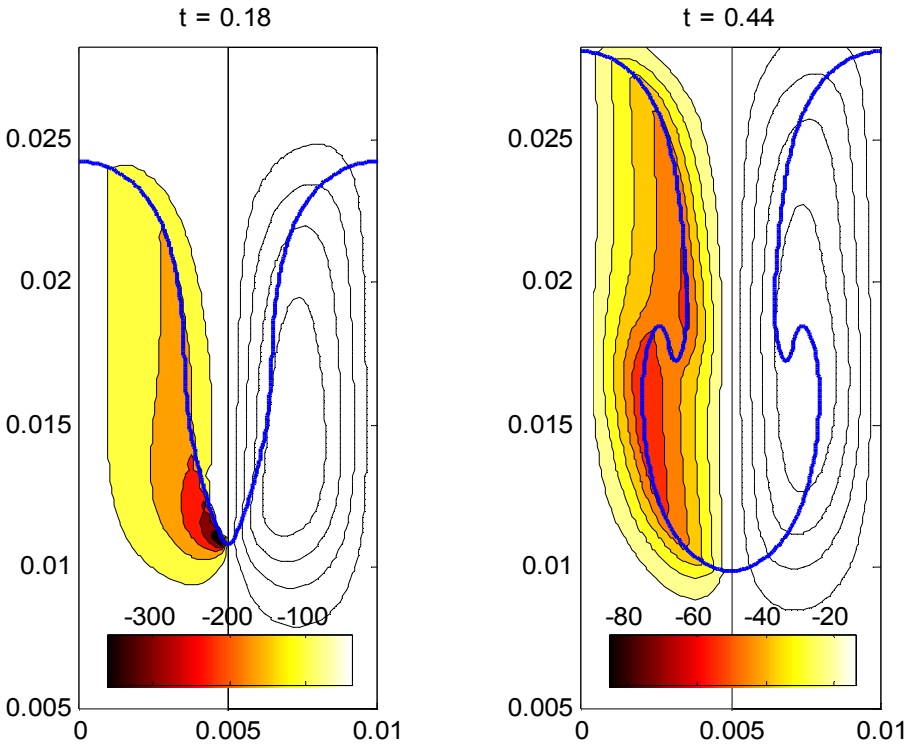


Figure 5. Flow detail of the spike for $A = 0.25$ at $t = 0.18$ (left); flow detail of the mushroom cap for $A = 0.815$ at $t = 0.44$ (right). The left and right sides of the pictures show the vorticity field and the streamfunction, respectively. The interface is represented by the blue line.

Broken Dam Problem. The broken dam problem is another test case problem which has been used to validate free-surface codes [13,19-22]. This problem consists in the collapse of a column of water on a dry bed. The fluid is initially at rest in a square reservoir of dimension H, when the walls are rapidly removed at the beginning of the simulation. The numerical results from simulations are usually compared with the experimental data of Martin and Moyce [23]. In our simulations, the slip boundary condition is assumed on the horizontal bed. Therefore, the flow, which is constantly accelerated by the gravity, does not experience any resistance. As a result, a thin layer of fluid, with a high velocity, develops at the front of the water wave. In fact, the only obstacle of the flow is the presence of the air. Therefore, the viscosity of the air is a critical numerical parameter. A small viscosity is necessary for the simulation to converge. However, its value should be as low as possible in order not to affect the flow of water. The problem is non-dimensionalized with the normalized time $t^* = t\sqrt{g/H}$ and the normalized dimension $x^* = x/H$ and $y^* = y/H$. The density of water and air were set to 1 and 0.001, respectively, while the normalized gravitational acceleration was equal to -1. For the viscosities, we used the values 0.001 and 0.0005. The adaptive time-step increment was automatically adjusted in order to keep a courant number of the simulation at 0.1. We perform the simulation with three different uniform meshes of 20×40, 30×60, and 60×120 control volumes. Fig. 6 shows snapshots of the evolution of the free-surface profile, obtained with the finest mesh resolution. Our results agree qualitatively well with other numerical simulations [19,21,22]. In Fig. 7, we also compared our predictions of the position of the front wave, and the height of the remaining column of water, with the experimental data Martin and Moyce [23]. Results for the height of the remaining column of water are in very good agreement with the experimental data, and no differences are observable between the calculations with different meshes. Concerning the position of the front wave, at a certain point of the simulation, the results become mesh-dependent. The finest mesh predicts farther displacements of the tip of the wave, whereas the coarsest mesh fits better the experimental data.

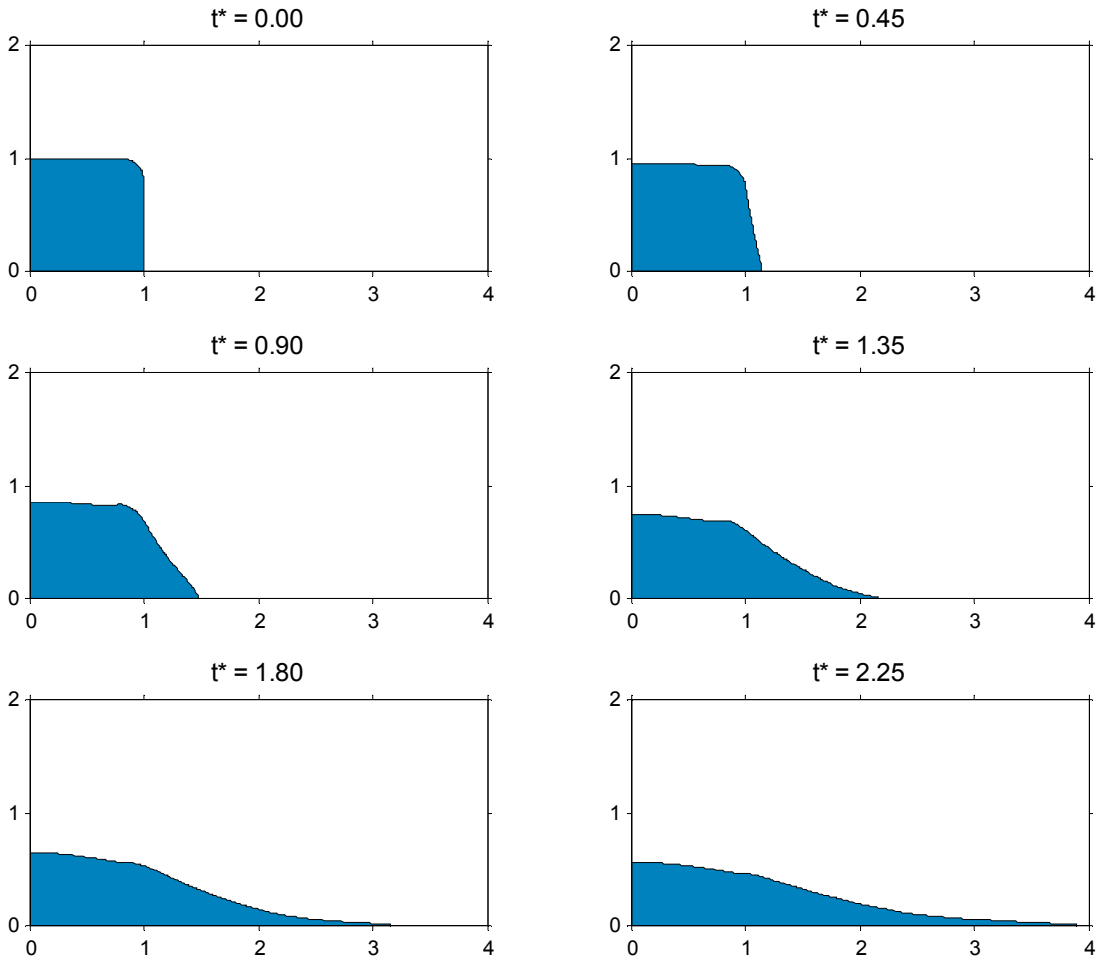


Figure 6. Snapshots of the free surface profile of the water at different times, calculated on the fine mesh. Time and space coordinates are expressed in nondimensionalized values.

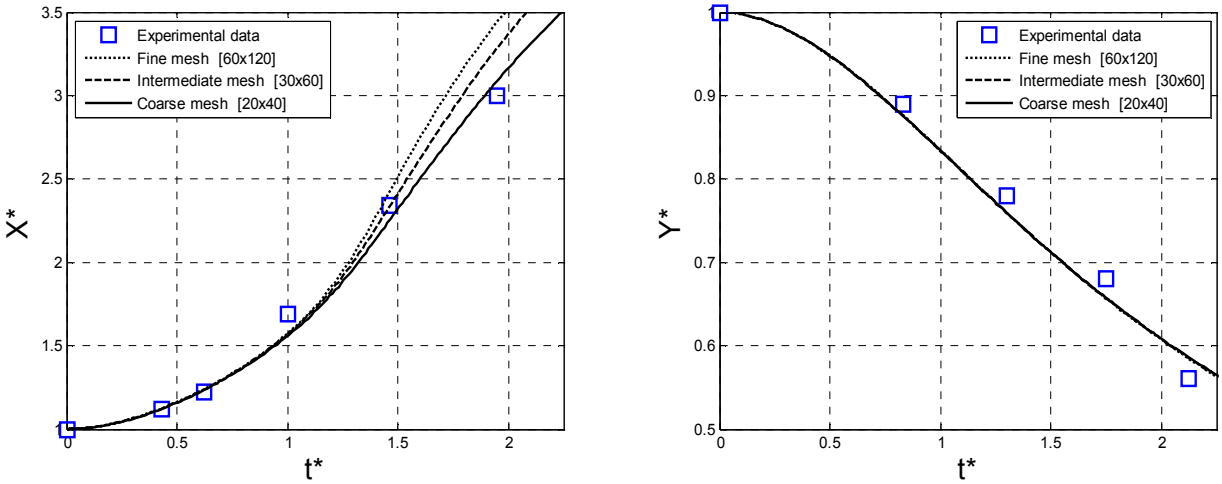


Figure 7. (left) Normalized position of the water front and (right) normalized height of the residual water column, as a function of time. Numerical results are compared with the experimental data of Martin and Moyce [23].

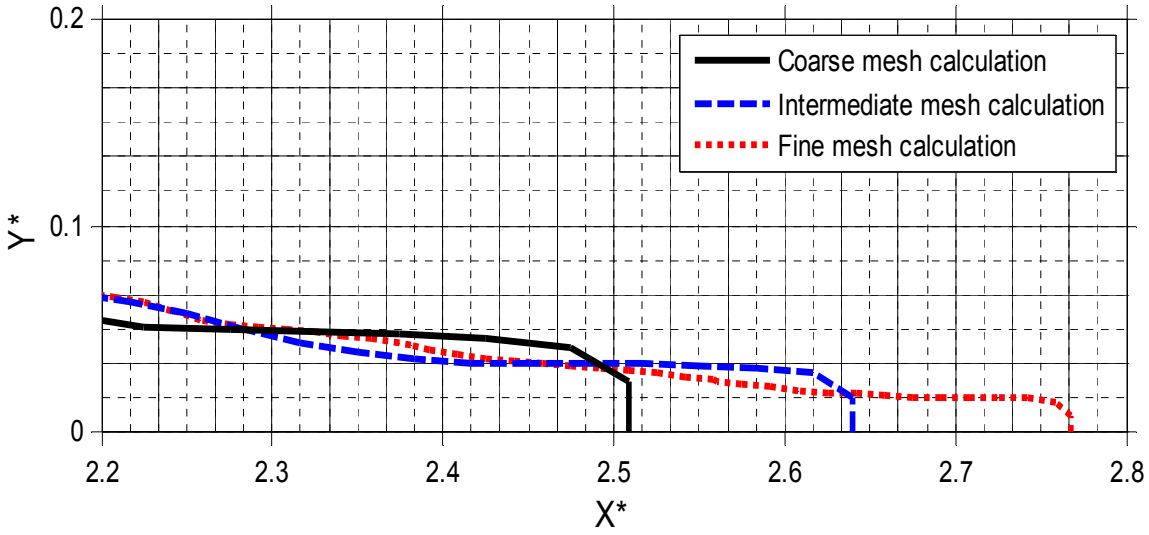


Figure 8. Close-up of the front wave predicted at $t^* = 1.60$ for the different meshes. The solid and dotted grids correspond respectively to the intermediate and fine dual meshes used for the calculation of the volume fraction.

This mesh-dependency can be explained by a close-up look at the wave tip. Fig. 8 depicts the wave tips predicted by the different meshes, where the dual control volumes used by the VOF algorithm are shown in the background (for the intermediate and fine meshes only). It shows that inside the cells closed to the wall, the interface is approximated by a vertical line, because of the slip condition (similar to symmetry). Consequently, the VOF method predicts a wave tip of the same thickness of the mesh size. In general, the VOF method cannot predict accurately the interfaces of the fluid films which have thicknesses thinner than the mesh resolution. However, in this simulation there are not any resistance forces to retain the flow of the wave tip (except the small viscosity of the air), as we use the slip boundary condition at the wall. As a consequence, in principle the simulation could predict an infinitely small thickness of the film, if the mesh was refined. In the real-life experiments, the thin film of water is influenced by the surface tension and the wetting angle (which have been neglected in the simulations). Apart from the small region of the wave tip, the positions of the free surface calculated with the different meshes are undistinguishable.

Summary

We have implemented a two-phase flow solver for 2D calculation, based on a pure streamfunction formulation and the volume-of-fluid method. The streamfunction formulation avoids the problem associated with the calculation of the pressure field at low Reynolds number. In addition, this scheme is more accurate than the classical fractional-step methods, because it does not generate splitting errors. Moreover, the mass conservation is fulfilled to the machine precision. The interface is tracked with the volume-of-fluid method. Two test case problems have been investigated, the Rayleigh-Taylor instability and the broken-dam problem. The growth rate of the Rayleigh-Taylor instability agrees very well with the theoretical growth rate in the range of small-amplitude perturbations. The numerical results for the broken dam problem fits well the experiments. We also showed the limit of the volume-of-fluid method in the prediction of very thin film of liquid, without considering the surface tension and the wetting angle. However, in the applications of polymer co-processing, these effects are negligible. In our future works, the model will be extended to non-Newtonian fluids, including materials with shear-thinning and viscoelastic rheological behaviors.

Acknowledgements

The authors would like to acknowledge the support of the Scientific Research Councils on Technology and Production Sciences (FTP) (Contract No. 09-072888, OPTIMAC), which is part of the Danish Council for Independent Research (DFF).

References

[1] J. Dooley, K.S. Hyun, K. Hughes, An experimental study on the effect of polymer viscoelasticity on layer rearrangement in coextruded structures, Polym. Eng. Sci. 38 (1998) 1060-1071.

[2] A.J. Chorin, Numerical solution of Navier-Stokes equation, Math. Comp. 22 (1968) 745-762.

[3] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys. 108 (1993) 51-58.

[4] M.-J. Ni, M.A. Abdou, A bridge between projection methods and SIMPLE type methods for incompressible Navier-Stokes equations, Int. J. Numer. Meth. Engng. 72 (2007) 1490-1512.

[5] S. Turek, A comparative study of time-stepping techniques for the incompressible Navier-Stokes equations: from fully implicit non-linear schemes to semi-implicit projection methods, Int. J. Numer. Methods Fluids 22 (1996) 987-1011.

[6] J.K. Dukowicz, Approximate factorization as a high order splitting for the implicit incompressible flow equations, J. Comput. Phys. 102 (1992) 336-347.

[7] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2001) 464–499.

[8] W. Chang, F. Giraldo, J.B. Perot, Analysis of the exact fractional step method, J. Comput. Phys. 180 (2002) 183-199.

[9] R. Kupferman, A central-difference scheme for a pure stream function formulation of the incompressible viscous flow, SIAM J. Sci. Comput. 23.1 (2001) 1-18.

[10] T.L. Chow, Mathematical Methods for Physicists: A concise introduction, Cambridge University Press, Cambridge, 2000.

[11] M.A. Alves, P,J. Oliveira, F.T. Pinho, A convergent and universally bounded interpolation scheme for the treatment of advection, Int. J. Numer. Methods Fluids 41 (2003) 47-75.

[12] J.E. Pilliod, E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2004) 465-502.

[13] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39.1 (1981) 201-225.

[14] J.B. Bell, D.L. Marcus, A second-order projection method for variable density flows, J. Comput. Phys. 101 (1992) 334.348.

[15] E.G. Puckett, A.S. Almgren, et al., A high-order projection method for tracking fluid interfaces in variable density incompressible flows, J. Comput. Phys. 130 (1997) 269-282,

[16] P.G. Drazin, Introduction to Hydrodynamic Stability, Cambridge University Press, Cambridge, 2002.

[17] B.J. Daly, Numerical study of two fluid Rayleigh-Taylor instability, Phys. Fluids 10.2 (1967) 297-307.

[18] G. Tryggvason, Numerical simulations of the Rayleigh-Taylor Instability, J. Comput. Phys. 75 (1988) 253-282.

[19] J. Wu, S.-T. Yu, B.-N. Jiang, Simulation of two-fluid flows by the least-squares finite element method using a continuum surface tension model, Int. J. Numer. Meth. Engng. 42 (1998) 583-600.

[20] T. Nakayama, M. Mori, An Eulerian finite element method for time-dependent free surface problems in hydrodynamics, Int. J. Numer. Methods Fluids 22 (1996) 175-194.

[21] S. Shin, W.I. Lee, Finite element analysis of incompressible viscous flow with moving free surface by selective volume of fluid method, Int. J. Heat Fluid Flow 21 (2000) 197-206.

[22] M.S. Kim, W.I. Lee, A new VOF-based numerical scheme for the simulation of fluid flow with free surface. Part I: New free surface-tracking algorithm and its verification, Int. J. Numer. Methods Fluids 42 (2003) 765-790.

[23] J.C. Martin, W.J. Moyce, Part IV. An experimental study of the collapse of liquid columns on a rigid horizontal plane, Philos. Trans. R. Soc. London, Ser. A 244.882 (1952) 312–324.

C. Paper–III

Comminal, R, Spangenberg, J & Hattel, JH 2015, 'Cellwise conservative unsplit advection for the volume of fluid method', *Journal of Computational Physics*, vol. 283, pp. 582–608.

Journal of Computational Physics 283 (2015) 582–608

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Cellwise conservative unsplit advection for the volume of fluid method

Raphaël Comminal [a,*], Jon Spangenberg [a,b], Jesper Henri Hattel [a]

[a] Department of Mechanical Engineering, Technical University of Denmark, Produktionstorvet, 2800 Kgs. Lyngby, Denmark
[b] Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ 08544, USA

ARTICLE INFO

Article history:
Received 26 May 2014
Received in revised form 27 November 2014
Accepted 1 December 2014
Available online 10 December 2014

Keywords:
Volume of fluid
Unsplit advection scheme
Cellwise approach
Pre-image backward tracing
Donating region
Volume conservation

ABSTRACT

We present a cellwise conservative unsplit (CCU) advection scheme for the volume of fluid method (VOF) in 2D. Contrary to other schemes based on explicit calculations of the flux balances, the CCU advection adopts a cellwise approach where the pre-images of the control volumes are traced backwards through the flow map. The donating regions of the fluxes are calculated via the streaklines of the grid intersections, represented as polygonal chains whose vertices are determined by backward tracing of particles injected in the flow at different times. High order accuracy is obtained from the fourth-order Runge–Kutta method, where intermediate velocities along pathlines are determined with quadratic temporal and bicubic spatial interpolations. The volumes of the donating regions are corrected in order to fulfill the discrete continuity of incompressible flows. Consequently, the calculation produces non-overlapping donating regions and pre-images with conforming edges to their neighbors, resulting in the conservativeness and the boundedness (liquid volume fraction inside the interval $[0, 1]$) of the CCU advection scheme. Finally, the update of the liquid volume fractions is computed from the intersections of the pre-image polygons with the reconstructed interfaces.

The CCU scheme is tested on several benchmark tests for the VOF advection, together with the standard piecewise linear interface calculation (PLIC). The geometrical errors of the CCU compare favorably with other unsplit VOF-PLIC schemes. Finally, potential improvements of the VOF method with the use of more precise interface representation techniques and the future extension of the CCU scheme to 3D are discussed.

1. Introduction

The simulation of free-surface/multiphase flows is an important branch of computational fluid dynamics, frequently applied to the simulation of many natural and industrial flows of various scales, ranging from microfluidics to hydrodynamics. The use of accurate surface-tracking techniques is crucial to simulate interfacial instabilities (e.g. Rayleigh–Taylor), and flows driven by surface tension, such as thread breakup, bubble raising, droplet dynamics, etc.

Common techniques for multiphase flow simulations include the *front-tracking* methods [1], the *volume-of-fluid* (VOF) method, and the *level-set* method [2,3]. While the front tracking methods represent explicitly the interface with geometrical objects (for instance markers, segment lines and planes), the VOF and level-set methods treat implicitly the interface, through an additional field variable, the color function. The main advantage of the implicit interface representations is that

* Corresponding author.
E-mail address: raphael.comminal@gmail.com (R. Comminal).

http://dx.doi.org/10.1016/j.jcp.2014.12.003
0021-9991/

they can handle arbitrary changes of topology [4]. Moreover, explicit interface representations require complex operations for the merge and breakup of interfaces; see for instance [4,5].

In the level-set method, the color function is the signed distance to the interface, governed by an advection-like equation. The advection equation is solved in an Eulerian framework using standard methods for partial differential equations, such as finite-differences, finite-volumes or finite-elements. The drawback of the level-set method is that it does not conserve the volume of each phase exactly.

In the VOF method, introduced by Hirt and Nichols in 1981 [6], the color function $\chi(\boldsymbol{x}, t)$ is an indicator of the material present at the position $\boldsymbol{x}$ and the time t:

$$\chi(\boldsymbol{x}, t) = \begin{cases} 1 & \text{if fluid no. 1 is present at } (\boldsymbol{x}, t), \\ 0 & \text{if fluid no. 2 is present at } (\boldsymbol{x}, t). \end{cases} \tag{1}$$

Sharp interfaces are defined at the jump discontinuities of the color function. The evolution of the color function is governed by a transport equation:

$$\frac{\partial \chi}{\partial t} + u_i \frac{\partial \chi}{\partial x_i} = 0. \tag{2}$$

On the discrete level, the average of the continuous color function over a control volume C gives the *liquid volume fraction* $f_C(t)$, i.e. the relative amount of fluid phase no. 1 inside C:

$$f_C(t) = \frac{1}{V_C} \int_C \chi(\boldsymbol{x}, t) \mathrm{d}\boldsymbol{x}, \tag{3}$$

where V_C is the total volume of C. The cells where $0 < f < 1$ contain an interface and they are hereafter referred to as the *interfacial cells*. Algebraic numerical methods for the resolution of hyperbolic conservation equations [7,8] are not suited to solve Eq. (2), because of their inherent numerical diffusion that smoothens the jump discontinuities. Consequently, algebraic advection schemes, such as the CICSAM [9], smear the sharp interfaces, resulting in a loss of precision and the non-conservation of the liquid volume fraction. The original VOF method rather uses a geometrical scheme to determine the fluxes of the liquid volume fraction through the edge of the control volume. The geometrical advection schemes require (i) an explicit geometrical reconstruction of the interface position in each interfacial cell, and (ii) an explicit evaluation of the donating region (DR) relative to each flux. The DR corresponds to the compact set of points that contains Lagrangian particles contributing to the flux, within a certain time interval, as defined in [10]. The intersection of the donating region with the reconstructed interface determines the amount of liquid volume that is advected through the edge of the control volume.

The standard technique to represent the interface is the piecewise linear interface calculation (PLIC) [11], in which the interface inside each interfacial cell is approximated by a segment, parameterized as

$$\hat{\boldsymbol{n}} \cdot \boldsymbol{x} + d = 0, \tag{4}$$

where $\hat{\boldsymbol{n}}$ is the normal vector of the interface pointing outward to the fluid phase no. 1, and where d is the signed distance of the interface to an origin. There are different methods to estimate the interface normal from the distribution of the liquid volume fraction, see Fig. 1. The simplest approximation is the normed gradient of the volume fraction [11] (Youngs' method)

$$\hat{\boldsymbol{n}} = -\frac{\nabla f}{|\nabla f|}, \tag{5}$$

evaluated by finite-differences using the values in the four neighboring cells; see Fig. 1(a). This gradient method is first-order accurate and does not reconstruct all line interfaces exactly.

Another method, the least-square VOF interface reconstruction algorithm (LVIRA), proposed by Puckett [12,13], is based on the minimization of the difference between the volume below the prolonged interface into the nearest neighboring cells, and the actual volume fraction in those cells, see Fig. 1(b). The minimization of the hatched area is a non-linear problem; therefore the LVIRA method requires an iterative optimization algorithm. An efficient variant of the original least-square method, the ELVIRA method [13], consists in evaluating only 12 candidates of the interface normal, from different backward, central and forward finite differences of the row and column sums of the liquid volume fractions in the 3×3 stencil of the nearest neighbors. Finally, the normal is chosen as the one of the 12 slopes that minimizes the least square error. Both LVIRA and ELVIRA are second-order accurate and reconstruct all line interfaces exactly.

An alternative method, the moment-of-fluid interface reconstruction algorithm [14], evaluates the interface normal without information of the liquid volume fraction in the neighboring cells. Instead, it uses the position of the center of mass of the liquid phase in the cell as additional information. The center of mass of the liquid phase in each cell is also advected with the flow, similarly as Lagrangian markers. The interface reconstruction algorithm consists in minimizing the distance δ between the reconstructed center of mass and the advected center of mass; see Fig. 1(c). The moment-of-fluid algorithm produces notable improvement in the reconstructions of sharp corners.

Finally, smoothening the variations in the curvature can enhance the accuracy of the interface reconstruction; see Fig. 1(d). The orientation of the PLIC is corrected in [15,16] based on spline interpolations of the slope along the interface. This method also provides a better estimation of the surface curvature, for the modeling of surface tension.

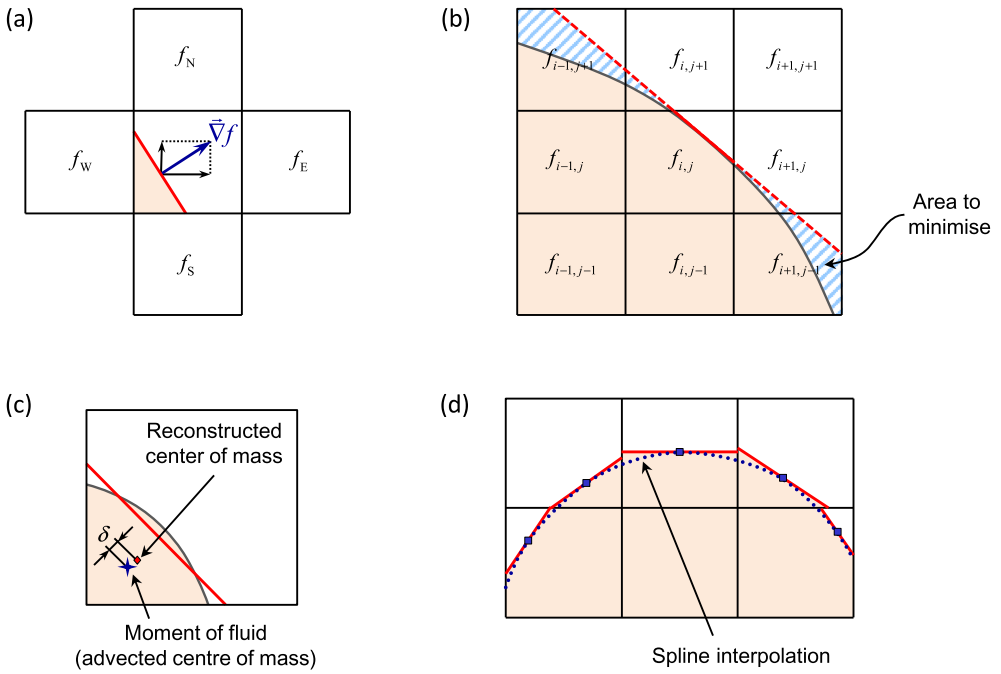


Fig. 1. Various techniques for piecewise linear interface calculation: (a) gradient method, (b) least-square minimization error, (c) moment-of-fluid method, (d) spline interpolation correction.

Once piecewise linear interfaces are reconstructed, the new liquid volume fractions are updated using a geometric advection algorithm, which evaluates the fluxed liquid volume from the geometric determination of the DR. The first generations of VOF algorithms were based on directional-split advection schemes, where the liquid volume is transported along one direction at a time; see Fig. 2(a). There are two approaches to propagate the liquid volume: the backward *Eulerian implicit* method, and the forward *Lagrangian explicit* method [5]. Many variants of the directional-split advection scheme have been presented in the literature. However, to the authors' knowledge, only two directional-split methods conserve the liquid volume exactly: the combined Eulerian Implicit–Lagrangian Explicit (EI–LE) method [17–19], and the Eulerian implicit method of Weymouth and Yue [20], which properly integrates the 1D dilatation terms (i.e. the stretching of the velocity field). Nonetheless, all the directional-split advection schemes suffer from numerical diffusion. The lack of accuracy generally produces geometrical errors that distort the interface [13], particularly in non-uniform flows such as in simple shear. In addition, non-conservative schemes also produce unphysical liquid volume fractions (outside the interval $[0, 1]$), requiring the use of local volume redistribution algorithms to remove the overshoots ($f > 1$) and undershoots ($f < 0$). Moreover, the directional-split methods require intermediate interface reconstructions between each 1D advection, and the Courant number is often limited to maximum 0.5. In contrast, unsplit (or multidimensional) schemes advect the liquid volume in all the directions at once, and do not require intermediate interface reconstructions. Unsplit advection schemes have been developed to improve accuracy and overcome numerical artifacts of the directional-split schemes. In addition, only the unsplit schemes can be generalized to unstructured mesh.

Early unsplit advection schemes have been designed by Puckett et al. [12,13], Rider and Kothe [21], Harvie and Fletcher (DDR scheme, acronym of defined donating region) [22], see Fig. 2(b)–(d). The Rider and Kothe scheme estimates the flux's DR as a trapezoid, whose angles are determined by the ratio of the face-centered velocities of the cell. Thus, the corresponding edges of the adjacent DRs are not collinear. This creates situations where the liquid volume in overlapping DRs is fluxed twice, see Fig. 3(a), while the liquid volume in gaps is not advected at all, see Fig. 3(b). Consequently, the Rider and Kothe scheme is not conservative, and suffer from numerical diffusion. In contrast, Puckett's and DDR schemes produce collinear edges of adjacent DR faces (without overlapping). Therefore, both of them conserve the volume, but only Puckett's scheme allows some volume to enter and exit a cell within a single time step. This volume is called the *en-route* material, and it should not contribute to the final liquid volume inside the cell. Being able to track en-route materials increases the accuracy of a scheme. Despite their variations, these three schemes belong to the same family, as the determination of the DR is solely based on the face-centered velocities.

Harvie and Fletcher [23] have proposed another method, the stream scheme, where the DR is determined as the union of *streamtubes* around the characteristics of discrete points laying on the cell's face, see Fig. 2(e). Although the Stream scheme is not conservative, it properly advects the en-route material through the cells.

More recently, López et al. proposed the edge-matched flux polygon advection (EMFPA) scheme [16], where the construction of the DR is only based on the characteristics of the face's extremities, using the grid vertex velocities, see Fig. 2(f).

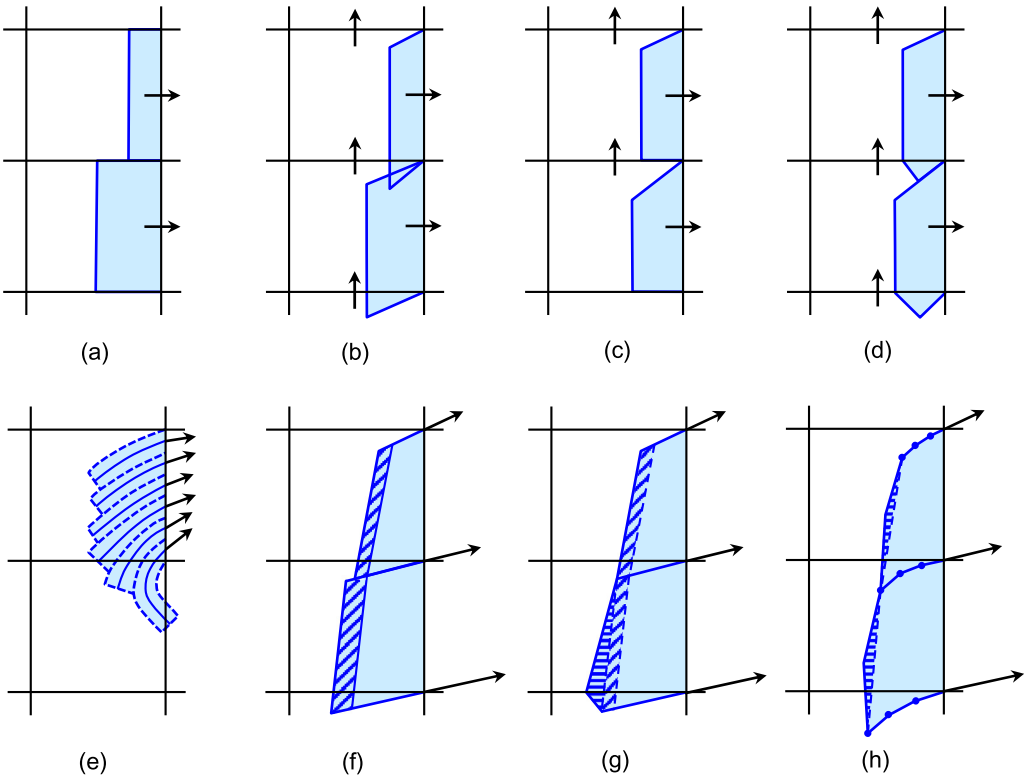


Fig. 2. Donating regions for various advection schemes: (a) directional-split, (b) Rider & Kothe, (c) DDR, (d) Puckett, (e) Stream, (f) EMFPA, (g) GPCA, and (h) CCU (proposed scheme).

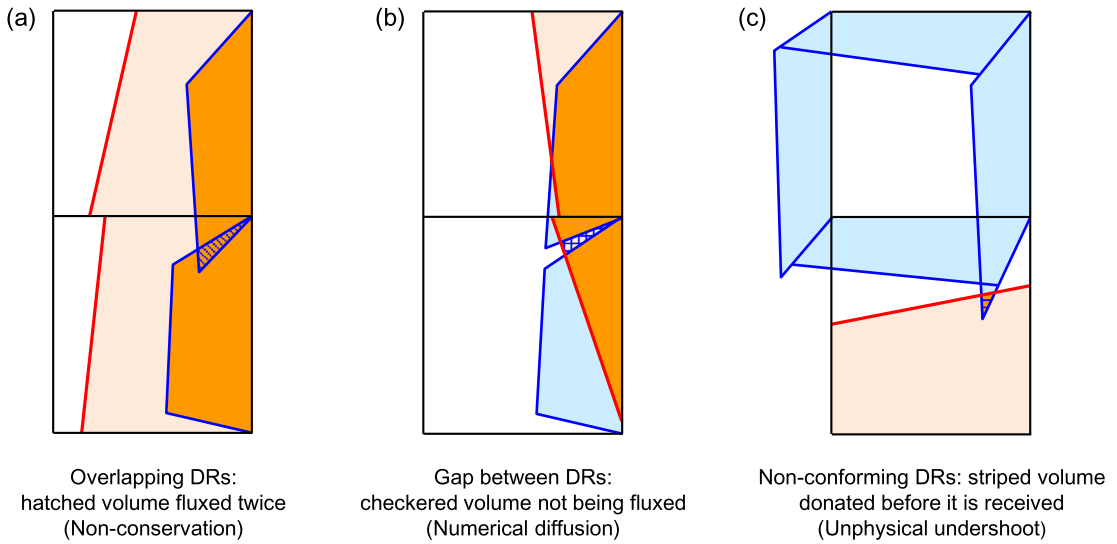


Fig. 3. Sources of numerical errors for the advection: (a) non-conservation of the liquid volume due to overlapping DRs, (b) numerical diffusion due to gaps between adjacent DRs, (c) undershot/overshot due to an inconsistency in the timelines, produced by non-conforming adjacent DRs.

The face-centered velocities are still used to compute a trapezoidal volume correction, which is added to the DR in order to make the scheme conservative. The volume corrections are represented in Fig. 2 by the hatched area. Mencinger and Zun have proposed a variant of the volume correction in [24]. The EMFPA scheme presents noticeable improvements of all previous schemes in the accuracy of the DR estimations. Yet, all advection schemes mentioned so far produce *non-conforming* adjacent DRs, in the sense that the common edges of adjacent DRs do not necessarily have the same lengths. It results in inconsistencies in the *timelines* of the grid's vertices. As shown in [24,25], this inconsistency can lead to situations where a cell donates a volume that it has not received yet, generating undershoots and overshoots. Thus, the previously mentioned

schemes may still produce unphysical liquid volume fractions, despite the global volume conservation; see Fig. 3(c). Unphysical results require the use of a local redistribution algorithm, as in [23] for instance. Such heuristic algorithms generally have a negative effect on the accuracy of the advection. Moreover, the occurrence of unphysical results is an indicator of intrinsic errors of the advection scheme.

The remapping method of Liovic and Lakehal [26] is an attempt to remedy the non-conformity of adjacent DRs, using an iterative—non-linear—remapping algorithm. However, as shown by its authors, this remapping method experiences convergence issues for flows with Courant numbers above 0.2, because of hourglass modes in the remapping. Cervone et al. [25] has proposed the geometrical predictor–corrector algorithm (GPCA), which performs an additional volume correction on top of the EMFPA scheme, in order to produce conforming DRs, see Fig. 2(g). Furthermore, Cervone et al. use a predictor–corrector scheme over three subdivisions of the time-step, rather than the forward Euler method as in the original EMFPA method.

Finally, Zhang [27] has presented a rigorous mathematical analysis of the VOF-advection through the prism of differential geometry and Boolean algebra. It demonstrates that the *edgewise* approach of the problem (based on the flux balance of the volume), is theoretically equivalent to a *cellwise* approach, where the liquid volume of the entire cell is retrieved by homeomorphism through the flow map. In practice, the boundaries of the cell are traced backward in time, delimiting the *pre-image* of the cell with respect to a certain time increment. By definition, the volume inside the pre-image will be the volume present inside the grid cell, after the considered time increment. While most of the VOF schemes adopt the edgewise viewpoint, the cellwise approach provides the bases for semi-Lagrangian volume-tracking methods, such as the polygonal area mapping (PAM) [28,29]. Zhang has also proposed an improved representation of the DR via donating region approximation by cubic splines (DRACS) [27,30], which has forth-order accuracy.

In this paper, we present a cellwise conservative unsplit (CCU) scheme for the VOF advection, improving the accuracy of current 2D unsplit schemes. The conservativeness and the boundedness of the CCU scheme are achieved through the construction of non-overlapping (with collinear edges) and conforming DRs. The method is composed of two stages. First, a non-conservative approximation of the cell's pre-image is computed via backward tracing of the cell's vertices with fourth-order Runge–Kutta integration. Secondly, a geometric volume flux correction is applied to enforce the volume conservation. At the end, the pre-image is represented as an 8-vertex polygon, which has exactly the same volume as its original cell. As an alternative to DRACS, the *streaklines* of the cell's vertices are represented for the DR calculation via polygonal chains rather than spline curves; see Fig. 2(h). Contrary to EMFPA, the CCU scheme maintains the conforming of the adjacent DRs. It also avoids the multi-stage predictor–corrector geometric correction procedure of the GPCA scheme, since CCU enforces the volume flux correction in a single step. Furthermore, the high-order accuracy in the backward tracing and DR representation reduces the magnitude of the volume flux correction needed, as compared to the other advection schemes.

Another originality of the CCU method is that the liquid volume fraction is updated based on the cellwise approach, which, in the authors' opinion, is easier to implement than the edgewise approach. Indeed, there is no explicit calculation of the flux balance of the liquid volume. The DRs are only used to compute the flux of the total volume (containing both material phases), but not the liquid volume (intersection of the DR with the PLIC). Unlike other schemes, the liquid volume fraction is solely evaluated inside the entire cell's pre-image. Consequently, the cellwise approach requires fewer calculations of polygonal intersections, which are more expensive than evaluating the volume of a polygon.

The paper is organized as follows: Section 2 provides a detailed description of the CCU advection scheme. Section 3 assesses the accuracy of the scheme in two classical benchmark problems, the deformed disks in the "single vortex in-a-box" and in the "4-vortex deformation field". The results are compared to those of other advection schemes. A discussion on further improvements of the VOF method with the use of more precise interface reconstruction algorithms, and on the potential extension of the CCU scheme to 3D is given in Section 4. An estimation of the computational cost of the CCU scheme is also provided. Finally the conclusions are summarized in the last section of the paper.

2. Cellwise conservative unsplit advection

At each time-step, the cells that are likely to undergo changes in their liquid volume fractions are tagged before the advection. These cells are the interfacial cells, plus N number of layers of neighboring cells, depending on the maximum Courant number. The algorithm only updates the liquid volume fraction of these tagged cells. The liquid volume fractions inside the other bulk cells are unchanged.

The CCU advection requires a brief recap of the analytic solutions of the VOF advection by Zhang [27]. The classical edgewise solution of the VOF advection—based on the flux balance—reads:

$$f_C\left(t^{n+1}\right) = f_C\left(t^n\right) + \frac{1}{V_C}\sum_{\text{edge}} \left\| \mathcal{D}_e^- \cap \mathcal{M}\left(t^n\right)\right\| - \left\| \mathcal{D}_e^+ \cap \mathcal{M}\left(t^n\right)\right\|, \tag{6}$$

where V_C is the total volume of the cell C, $\mathcal{M}(t^n)$ is the volume occupied by the liquid phase at the time t^n, $\mathcal{D}_e^-$ and $\mathcal{D}_e^+$ are the received and donated volumes, respectively, through the edge e during the time increment $\Delta t = t^{n+1} - t^n$. It has been demonstrated in [27] that, since the flow map is a homeomorphism, the edgewise solution (6) is equivalent to the following cellwise solution:

$$f_C\left(t^{n+1}\right) = \frac{1}{V_C}\left\| \overleftarrow{\mathcal{P}_{\Delta t}(C)} \cap \mathcal{M}\left(t^n\right)\right\|, \tag{7}$$

where $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is the pre-image of C, traced backward within the time increment Δt. The CCU advection scheme computes the new liquid volume fraction $f_C(t^{n+1})$ from the formula (7). The volume $\mathcal{M}(t^n)$ occupied by the liquid phase is calculated with the interface reconstruction algorithm. The main task of the CCU scheme is the determination of the pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ of the cell. It is performed in two steps:

i. First, the positions of the cell's vertices are traced backward in time with a 4th-order accurate Runge–Kutta integration. The intermediate velocities are evaluated with bicubic spline interpolations in space, and quadratic spline interpolations in time. The pre-images of the four cell vertices form a polygon, which is a first approximation of the cell's pre-image, yet non-conservative.
ii. Four additional vertices are added to the definition of the pre-image polygon, in order to adjust its volume without changing the position of four vertices obtained via backward tracing. The geometric volume flux correction uses explicit formulas—rather than iterations—based on the DR calculation.

Finally, the new liquid volume fraction is obtained from the intersection of the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ with the reconstructed PLIC. Each of these steps is described hereinafter.

2.1. Backward-tracing

The accuracy of the CCU scheme lies in the backward tracing of the cell's pre-image. We consider a rectangular grid cell C defined by its vertices 1, 2, 3, 4. Vertices A, B, C, D are obtained from the backward tracing of 1, 2, 3, 4, respectively. The polygon ABCD is the first approximation of the cell's pre-image, as shown in Fig. 4(a).

The backward tracing of a vertex is obtained with the reversed time-integration of the instantaneous velocity along the pathline of a Lagrangian particle p, initially located at the vertex's position. The instantaneous velocity of the particle p whose position $\boldsymbol{x}_p(t)$ varies in time is

$$\boldsymbol{v}(p,t) = \frac{d\boldsymbol{x}_p}{dt}. \tag{8}$$

The pre-image of the vertex is

$$\boldsymbol{x}_p\left(t^n\right) = \boldsymbol{x}_p\left(t^{n+1}\right) - \int_{t^n}^{t^{n+1}} \boldsymbol{v}\left(p,t^*\right)\mathrm{d}t^*. \tag{9}$$

In our scheme, the integral in (9) is solved numerically with the explicit 3/8 forth-order Runge–Kutta method:

$$\begin{aligned}
&\boldsymbol{x}_p\left(t^n\right) = \boldsymbol{x}_p\left(t^{n+1}\right) - \frac{\Delta t}{8}(\boldsymbol{v}_1 + 3\boldsymbol{v}_2 + 3\boldsymbol{v}_3 + \boldsymbol{v}_4) + \underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^5\right),\\
&\boldsymbol{v}_1 = \boldsymbol{u}\left(t^{n+1}, \boldsymbol{x}_p\left(t^{n+1}\right)\right),\\
&\boldsymbol{v}_2 = \boldsymbol{u}\left(t^{n+2/3}, \boldsymbol{x}_p\left(t^{n+1}\right) - \Delta t\boldsymbol{v}_1/3\right),\\
&\boldsymbol{v}_3 = \boldsymbol{u}\left(t^{n+1/3}, \boldsymbol{x}_p\left(t^{n+1}\right) + \Delta t(\boldsymbol{v}_1/3 - \boldsymbol{v}_2)\right),\\
&\boldsymbol{v}_4 = \boldsymbol{u}\left(t^{n}, \boldsymbol{x}_p\left(t^{n+1}\right) - \Delta t(\boldsymbol{v}_1 - \boldsymbol{v}_2 + \boldsymbol{v}_3)\right),
\end{aligned} \tag{10}$$

where $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$, $\boldsymbol{v}_4$ are intermediate velocities of the particle, and $\boldsymbol{u}(t,\boldsymbol{x})$ is the Eulerian velocity field. The VOF method is typically coupled with an implicit (or semi-implicit) Navier–Stokes solver, which resolves $\boldsymbol{u}(t,\boldsymbol{x})$ on a discrete set of locations, at discrete points in time. The values of the intermediate velocities are obtained from the temporal and spatial interpolations of the known discrete velocities.

In order to avoid velocity overshoots and to promote numerical stability, it is preferable to first solve the new velocity field implicitly, prior to solving the new liquid volume fraction explicitly, as in [31] for instance. The velocities at the intermediate times $t^{n+1/3}$ and $t^{n+2/3}$ are determined by quadratic spline interpolation within the time interval $t \in [t^n, t^{n+1}]$:

$$\boldsymbol{u}(t) = \boldsymbol{u}\left(t^n\right) + \left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\left(t - t^n\right) + \left[\boldsymbol{u}\left(t^{n+1}\right) - \boldsymbol{u}\left(t^n\right) - \left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\Delta t\right]\left(\frac{t - t^n}{\Delta t}\right)^2, \tag{11}$$

where the acceleration $\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}$ is known from the previous time-step. The acceleration at the end of the current time-step is evaluated and saved for use at the next time-step:

$$\left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^{n+1}} = \left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n} + 2\left[\frac{\boldsymbol{u}(t^{n+1}) - \boldsymbol{u}(t^n)}{\Delta t} - \left.\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t}\right|_{t^n}\right]. \tag{12}$$

The temporal quadratic approximation for the evolution of the velocity provides a balance between accuracy and efficiency. In fact, the spatial variations of the velocity field are more significant for the interpolation of intermediate velocities than

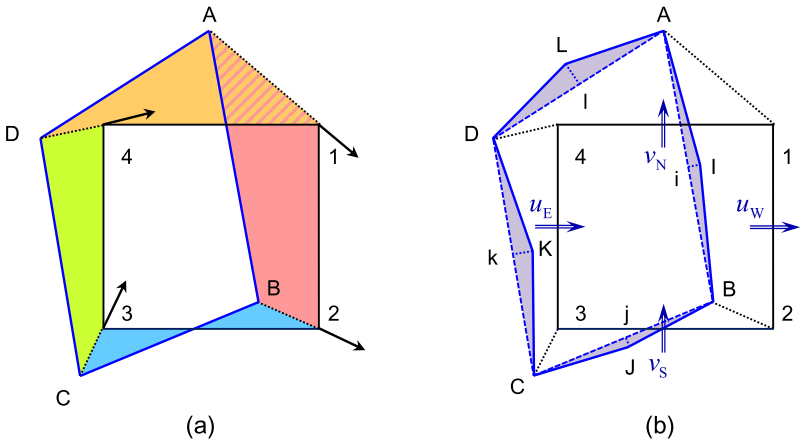


Fig. 4. Calculation of the pre-image polygon: (a) non-conservative 4-vertex approximation via backward tracing of the cell's vertex, (b) conservative 8-vertex polygon after the geometric volume flux correction.

the temporal variations. Cervone et al. [25] have used a simple predictor–corrector scheme with three subdivisions of the time-step, in conjunction with bilinear spatial interpolations, and reported that numerical errors in the integration of Eq. (9) can be as high as 10% of the distance $\|\boldsymbol{x}_p(t^{n+1}) - \boldsymbol{x}_p(t^n)\|$. They identified two main contributions to the total integration error: the interpolation error and the discretization error. The interpolation error comes from the bilinear spatial interpolations of the velocity field. The discretization error refers to the approximation of the cell's vertex velocities from an arithmetic average of the face-centered velocities. Thus, both main contributions to the backward tracing errors in [25] arise in reality from the linear approximation of the spatial variations of the velocity field. In our scheme, the accuracy of the backward tracing is enhanced by the use of bicubic spatial interpolations for the intermediate velocities:

$$\boldsymbol{u}(\xi,\eta) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}\xi^i\eta^j, \tag{13}$$

where ξ and η are the normalized local coordinates inside the cell, and a_{ij} are the 16 coefficients of the different terms, see Appendix A for more details. As a result, the interpolated velocity field is C^1-continuous and varies smoothly in both time and space. In addition, the quadratic temporal and bicubic spatial interpolations allow the velocity field to have local extrema in between the known discrete values.

Finally the positions of the vertices A, B, C, D are determined using formulas (10)–(13). The resulting polygon ABCD is an overall good approximation of the cell's pre-image, yet it does not fulfill the volume conservation (its volume is not strictly equal to the volume of C).

2.2. Geometric volume flux correction

The volume conservation of the cell is enforced by adjusting the area of its pre-image polygon. The positions of the vertices A, B, C, D, computed with overall fourth-order accuracy, are kept unchanged in order to have conforming adjacent DRs. Instead, four additional vertices I, J, K, L are introduced to the definition of the pre-image polygon, see Fig. 4(b). They are initially positioned at the midpoints i, j, k, l of the pre-image's edges [AB], [BC], [CD] and [DA], respectively. The area of the pre-image polygon AIBJCKDL is modified by adjusting the position of the vertices I, J, K, L along the median line of the pre-image's edges. Therefore, the volume corrections are added or removed to the pre-image's edges as isosceles triangles. The suitable positions of the vertices I, J, K, L are calculated via the geometric volume flux correction procedure.

The volume conservation is a consequence of the incompressibility constraint, which is the kinematic condition of solenoidal velocity fields. Integrating the divergence of a solenoidal velocity field over the control volume C and applying the divergence theorem yields:

$$\oiint_{\partial C} (\boldsymbol{u}\cdot\hat{\boldsymbol{n}})\mathrm{d}s = 0. \tag{14}$$

It translates into the continuity equation:

$$\sum_e A_e u_e = 0, \tag{15}$$

where e is the index of the edges, A_e are the lengths of the edges, and u_e the face-averaged velocities. The quantity

$$\Phi_e = A_e u_e \tag{16}$$

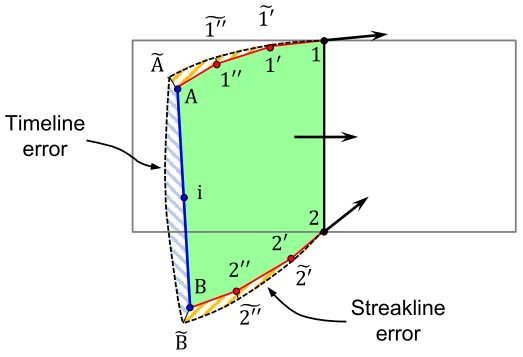


Fig. 5. Calculation of the DR of the edge 12. The polygonal chains 1–1′–1″–A and 2–2′–2″–B, and the segments A–i–B are the approximate representations of the streaklines and timeline, whose exact positions are represented by the black dashed curves. The hatched areas represent geometrical errors of the DR construction.

is the volume flux through the edge e. By virtue of the equivalence between the cellwise and edgewise approaches, an easy and robust way to enforce the volume conservation on the pre-image is to make the net volume fluxes of the DRs equal to the divergence-free volume fluxes Φ_e.

The calculation of the DR requires the evaluation of the *timelines* of the cell's faces, and the *streaklines* of the cell's vertices [10,30], see Fig. 5. Note that in transient flows, the streaklines are different from the pathlines calculated in the backward tracing, yet they have the same extremities. A streakline is the trace of particles continuously injected at a certain location, while a pathline is the trajectory of an individual particle. In our scheme, two discrete points $1'$ and $1''$ of the streaklines are calculated via backward tracing of particles injected in the flow at the cell's vertices, at the intermediate times $t^{n+1/3}$ and $t^{n+2/3}$. In order to avoid further interpolations of the velocity field, the position of the point $1'$ is obtained by the two-level Heun's method:

$$\begin{aligned}
&\boldsymbol{x}_p\left(t^n\right) = \boldsymbol{x}_p\left(t^{n+1/3}\right) - \frac{\Delta t}{6}(\boldsymbol{v}_1 + \boldsymbol{v}_2) + \underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^3\right),\\
&\boldsymbol{v}_1 = \boldsymbol{u}\left(t^{n+1/3}, \boldsymbol{x}_p\left(t^{n+1/3}\right)\right),\\
&\boldsymbol{v}_2 = \boldsymbol{u}\left(t^n, \boldsymbol{x}_p\left(t^{n+1/3}\right) - \Delta t\,\boldsymbol{v}_1/3\right),
\end{aligned} \tag{17}$$

and the position of the point $1''$ is obtained by the Kutta's three-level method:

$$\begin{aligned}
&\boldsymbol{x}_p\left(t^n\right) = \boldsymbol{x}_p\left(t^{n+2/3}\right) - \frac{\Delta t}{9}(\boldsymbol{v}_1 + 4\boldsymbol{v}_2 + \boldsymbol{v}_3) + \underset{\Delta t\to 0}{\mathrm{O}}\left(\Delta t^4\right),\\
&\boldsymbol{v}_1 = \boldsymbol{u}\left(t^{n+2/3}, \boldsymbol{x}_p\left(t^{n+2/3}\right)\right),\\
&\boldsymbol{v}_2 = \boldsymbol{u}\left(t^{n+1/3}, \boldsymbol{x}_p\left(t^{n+2/3}\right) - \Delta t\,\boldsymbol{v}_1/3\right),\\
&\boldsymbol{v}_3 = \boldsymbol{u}\left(t^n, \boldsymbol{x}_p\left(t^{n+2/3}\right) + \Delta t(2\boldsymbol{v}_1 - 4\boldsymbol{v}_2)/3\right).
\end{aligned} \tag{18}$$

Note that the variable Δt in the formulas (17) and (18) is the time increment of the whole time-step, not the time interval of the Heun or the Kutta's methods. Once the points $1'$ and $1''$ are determined, the DR is represented by the polygon $11'1''\mathrm{AiB}2''2'2$. As shown in [27], the error in the calculation of the DR has two contributions. The first contribution comes from the integration errors in the backward tracing using the formulas (10), (17), and (18), i.e. the calculated positions of points $1', 1'', \mathrm{A}$ are different from their theoretical positions $1', 1'', \mathrm{A}$. The second error contribution is due to the reconstructed representation of the streaklines and timelines, as for instance the approximation of curved lines with straight lines as in [16,24,25]. For higher precision, the streaklines are represented in our scheme with polygonal chains 1–1′–1″–A. The use of three sub-segments provides a balance between accuracy and efficiency.

The net volume flux F_e through a cell's edge e is calculated via the area of its DR polygon. For a regular—non-self-intersecting—DR polygon $\mathcal{P}_e$, depicted in Fig. 6(a), the net volume flux is simply its area, computed with the following formula:

$$F_e = \|\mathcal{P}_e\| = \frac{1}{2}\left|\sum_{i=0}^{n-1} x_i y_{i+1} - x_{i+1} y_i\right|, \tag{19}$$

where n is the number of vertices defining the polygon $\mathcal{P}_e$, and (x_i, y_i) are their coordinates. The formula (19) is valid for both convex and concave polygons. Furthermore, particular attention must be paid to the case of irregular—self-intersecting—polygons. We distinguish between two cases of self-intersections in the DR polygons, based on whether they involve the cell's edges or not. The CCU scheme only allows self-intersections involving the cell's edges. As pointed out in [27], the ability of an advection scheme to properly deal with this event is crucial for its accuracy. Directional-split, and

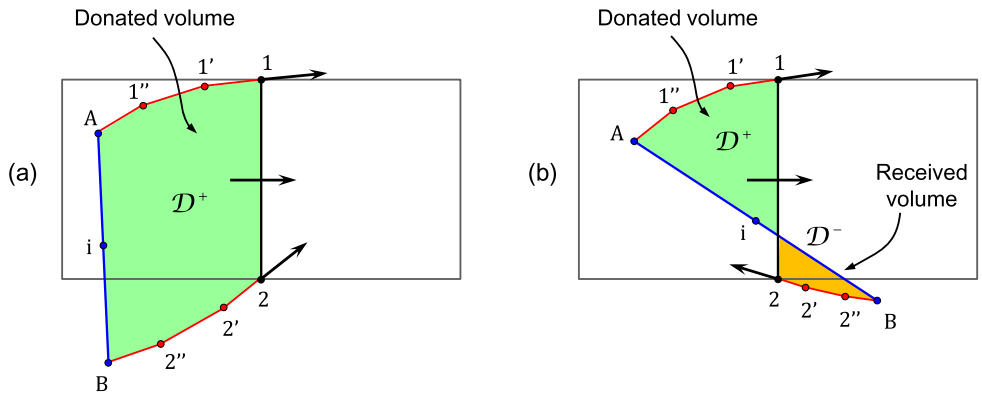


Fig. 6. Calculation of the volume flux through the edge 12, (a) in the case of a regular DR polygon, (b) in the case of an admissible self-intersecting DR polygon.

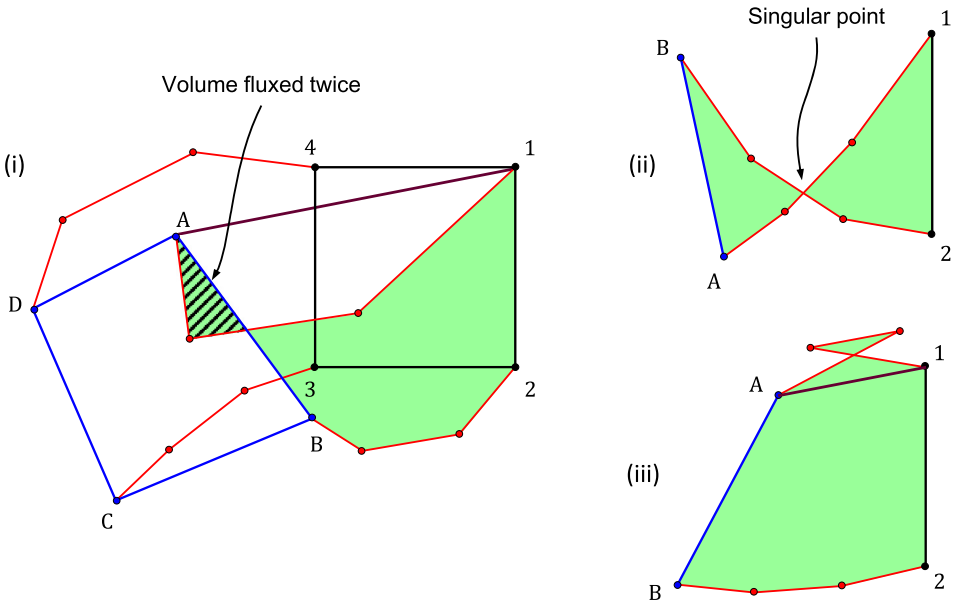


Fig. 7. The three scenarios of non-admissible irregular DR polygons (not involving self-intersection with the edge 12).

early unsplit advection schemes [12,13,17–22] are not able to take into account both entering and exiting volumes that may cross an edge within the same time increment. The CCU scheme differentiates the entering and exiting volumes of an edge e, by splitting the admissible irregular DR polygon $\mathcal{P}_e$ into two entities $\mathcal{D}_e^+$ and $\mathcal{D}_e^-$, whose contributions to the net volume flux have opposite signs, such that

$$\begin{aligned} &\mathcal{D}_e^+ \cup \mathcal{D}_e^- = \mathcal{P}_e, \\ &\mathcal{D}_e^+ \cap \mathcal{D}_e^- \subset e. \end{aligned} \tag{20}$$

The areas of $\mathcal{D}_e^+$ and $\mathcal{D}_e^-$ are the donated and received volumes of the cell C through the edge e, respectively, see Fig. 6(b). Then, the net volume flux of an admissible irregular DR through the edge e is given by:

$$F_e = \left\| \mathcal{D}_e^- \right\| - \left\| \mathcal{D}_e^+ \right\|. \tag{21}$$

The case of non-admissible irregular DR polygons is now examined. There are three scenarios of non-admissible self-intersections: (i) one of the streakline polygonal chains intersecting the timeline, (ii) the two polygonal chains intersecting each other's, or (iii) one of the polygonal chains self-intersecting, see Fig. 7. Scenario (i) violates the homeomorphism of the flow map. As a result, the hatched area inside the pre-image polygon erroneously contributes to the liquid volume fraction of the cell, while it is fluxed twice to different cells. If not patched, this situation causes the non-conservation of the liquid volume, and for that reason the problematical polygonal chain is replaced by the straight segment [1A]. Scenario (ii) is also problematic because it is very likely to produce irregular pre-image polygons that would also result in non-conservation of the liquid volume. However, this scenario involves a singular point in the velocity field, which is very unlikely to occur in

physical incompressible flows. Therefore, this scenario is not patched. The last scenario (iii) does not breakdown the conservativeness, but it makes the evaluation of the DR's area more difficult since the formula (19) does not hold for irregular polygons. Consequently, we also choose in this case to replace the self-intersecting polygonal chain by the straight segment [1A]. Other choices are possible; nevertheless the situation of self-intersecting streaklines is very rare. In fact, these three scenarios of non-admissible self-intersecting DR are more likely to be numerical artifacts arising from numerical errors, than actual features of the real flow. Therefore, another way to prevent them is to keep the maximum Courant number $C_{\max}$ to reasonable values. An idea is to adapt the maximum Courant number $C_{\max}$, using a criterion based on the volume conservation error of the 4-vertex polygonal approximation of the pre-image.

Finally, the amount of volume flux correction that the DR requires is given by the difference

$$\Delta\phi_e = \Phi_e - F_e. \tag{22}$$

The needed volume flux correction is added or removed to the area of the DR by moving the position of the vertex I from the midpoint i of the segment [AB], by the distance h_e along the bisector line of [AB], enforcing systematic conservativeness, see Fig. 4(b). The volume flux corrections are the isosceles triangles colored in purple, thus

$$h_e = 2\Delta\phi_e / \|\mathrm{AB}\|. \tag{23}$$

The algorithm is valid for all the topologies of admissible DRs: convex, concave and self-intersecting, the three of them being represented in Fig. 4(b). After application of the volume flux correction to all the DRs, the resulting 8-vertex polygon AIBJCKDL is a conservative representation of the cell's pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$.

2.3. Cellwise VOF update

The last step of the cellwise advection consists in computing the intersection of the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ with the area $\mathcal{M}(t^n)$ occupied by the liquid material. $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is determined by the CCU scheme while $\mathcal{M}(t^n)$ is obtained from the PLIC interface reconstruction. The intersection of polygons is a Boolean operation that requires algorithms of *combinatorial computational geometry*. For an introduction on the topic, we refer to the handbook of O'Rourke [32]. Libraries of Boolean operations on polygons are available in several programming languages. The MATLAB® Mapping Toolbox provides the polybool function for Boolean operations on arbitrary polygons.

Alternatively, the intersection $\overleftarrow{\mathcal{P}_{\Delta t}(C)} \cap \mathcal{M}(t^n)$ can be computed using a simple line/line intersection algorithm. A search algorithm is first used to find all the underlying grid cells with potential intersection with $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$. These underlying cells have at least one of their vertices within the interval $[\min(x_{\overleftarrow{\mathcal{P}(C)}}), \max(x_{\overleftarrow{\mathcal{P}(C)}})] \times [\min(y_{\overleftarrow{\mathcal{P}(C)}}), \max(y_{\overleftarrow{\mathcal{P}(C)}})]$, bounded by the minimum and maximum values of the coordinates of the pre-image polygon's vertices. Then, the pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ is split into its intersections with each of the underlying grid cells, see Fig. 8. The different cuts of $\overleftarrow{\mathcal{P}_{\Delta t}(C)}$ are obtained using the TruncatePolygonByLine algorithm, described below. The liquid volume inside each cut of the pre-image polygon is obtained by further truncation with the PLIC line. Finally, the cellwise update of the liquid volume fraction is recovered by summing up the liquid volume contributions from all the truncations of the pre-image polygon, corresponding to the area colored in purple in Fig. 8.

Algorithm TruncatePolygonByLine: Compute the truncation of a polygon by an oriented line.

```
1:  function TruncatePolygonByLine(Q, n̂, d)
2:    input Q                                   List of the polygon's vertices {Q_1, Q_2 ...}
3:    input n̂                                   Normal vector of the oriented line
4:    input d                                   Distance to origin of the oriented line
5:    if n̂ · Q_1 + d > 0 then in_1 ← false      The first polygon's vertex is outside the cut
6:    else in_1 ← true end if                   The first polygon's vertex belongs to the cut
7:    if in_1 then P ← {Q_1}                    Save the first polygon's vertex in the output list
8:    else P ← {∅} end if                       Create an empty output list
9:    for k = 2 → NumberOfElements(Q) do        Loop over the polygon's edges
10:     if n̂ · Q_k + d > 0 then in_2 ← false    The k^th polygon's vertex is outside the cut
11:     else in_2 ← true end if                 The k^th polygon's vertex belongs to the cut
12:     if in_1 xor in_2 then                   Evaluate if the edge crosses the oriented line
13:       n̂_2 ← (Q_k − Q_{k−1})^⊥ / ‖Q_k − Q_{k−1}‖   Calculate the normal vector of the edge
14:       d_2 ← Q_k · n̂_2                       Calculate the distance to origin of the edge
15:       s ← ComputeLinesIntersection(n̂, d, n̂_2, d_2)   Compute the intersection of the two lines
16:       if s ∈ [Q_k Q_{k−1}] and (s ≠ Q_k and s ≠ Q_{k−1}) then
17:         P ← {P, s} end if                   Add the intersection point to the output list
18:     end if
19:     if in_2 then P ← {P, Q_k} end if        Add the k^th polygon's vertex to the output list
20:     in_1 ← in_2
21:   end for
22:   return P                                  Return the vertex list of the truncated polygon
23: end function
```

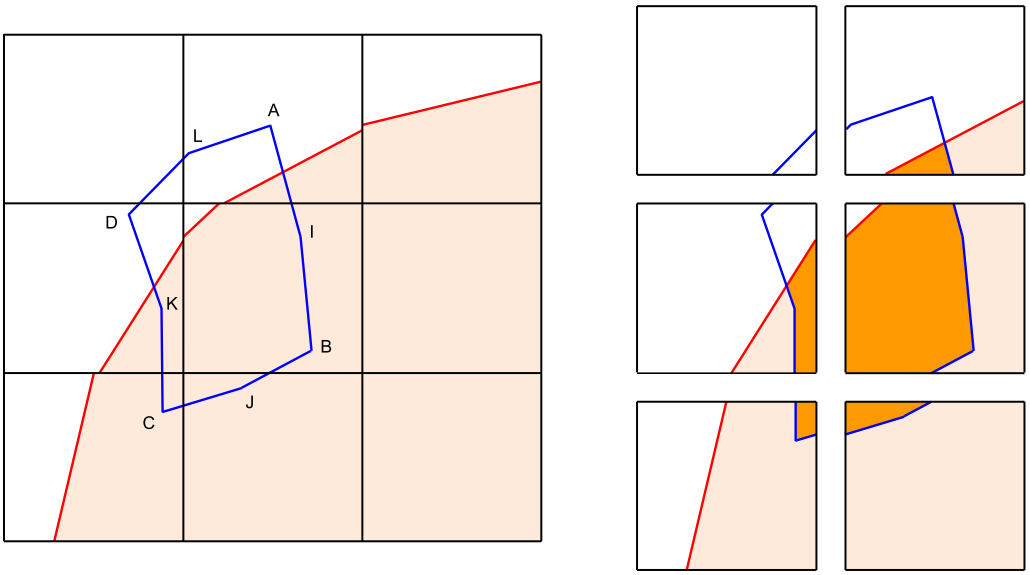


Fig. 8. Truncation of the pre-image polygon AIBJCKDL with the underlying grid cells and their reconstructed PLIC lines. The liquid material occupies the colored area. The dark area represents the final intersection of the pre-image with the liquid material.

3. Numerical tests

In this section, the accuracy of the CCU scheme is assessed on classical benchmark tests. Firstly, we define a measure of the pre-image calculation error, in order to differentiate between pure advection error and interface reconstruction error. Secondly, the CCU scheme is tested on two benchmark problems involving non-uniform deformations: the *single vortex in-a-box* and the *4-vortex deformation field* test cases. Our results are compared with those of other advection schemes reviewed in the introduction. For the sake of comparison with other publications, the simulations have only been performed on structured Cartesian grids. We use an adaptive time-stepping procedure, which assumes first-order temporal variations of the velocity; see Appendix B.

3.1. Pre-image calculation errors

The pre-images construction error E_1 is defined as the area of the difference (exclusive OR) between the computed 8-vertex pre-image polygon $\overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{CCU}}$ and the exact pre-image $\overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{exact}}$, summed for all the grid cells C:

$$E_1 \equiv \sum_C \left\| \overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{CCU}} \oplus \overleftarrow{\mathcal{P}_{\Delta t}(C)}^{\mathrm{exact}} \right\|. \tag{24}$$

In our numerical tests, the "exact" pre-image of each cell is a very fine polygonal representation with 80 vertices, whose positions are obtained by numerical integration with a predictor–corrector scheme over 20 sub-divisions of the time-step. The pre-image calculation error E_1 is very relevant to assess the accuracy of cellwise advection schemes, as the liquid volume fraction is obtained from the intersection of the interface with the area of whole cell's pre-image. In reality, E_1 is an upper bound of the total advection error, independent of the interface position and the interface reconstruction, i.e. if all the cells were affected by the update of the liquid volume fraction and if the interface was represented exactly. Moreover, E_1 is zero when the pre-images are exactly constructed, thus it is a norm.

The pre-image calculation error produced by the CCU scheme is measured for various incompressible flows within the rectangular calculation domain $[0,1]\times[0,1]$. In the tests, the velocities at the cell's vertices are derived from analytical expressions of the streamfunction ψ, with $u = \mathrm{d}\psi/\mathrm{d}y$ and $v = -\mathrm{d}\psi/\mathrm{d}x$. The face-averaged velocities are obtained by the difference in the streamfunction level between extremities of the edges. The following flows are considered:

- the simple translation field: $\psi = x + y$,
- the simple rotation field: $\psi = (x-0.5)^2 + (y-0.5)^2$,
- the single vortex in-a-box field: $\psi = -\frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)$,
- the 4-vortex deformation field: $\psi = \frac{1}{4\pi}\sin(4\pi(x+0.5))\cos(4\pi(y+0.5))$.

These velocity fields have symmetries and periodicities. Fig. 9 shows the computed versus "exact" pre-images, as well as the streaklines of the grid vertices. Note that Fig. 9 only displays representative unit cells of the flows, for relatively coarse

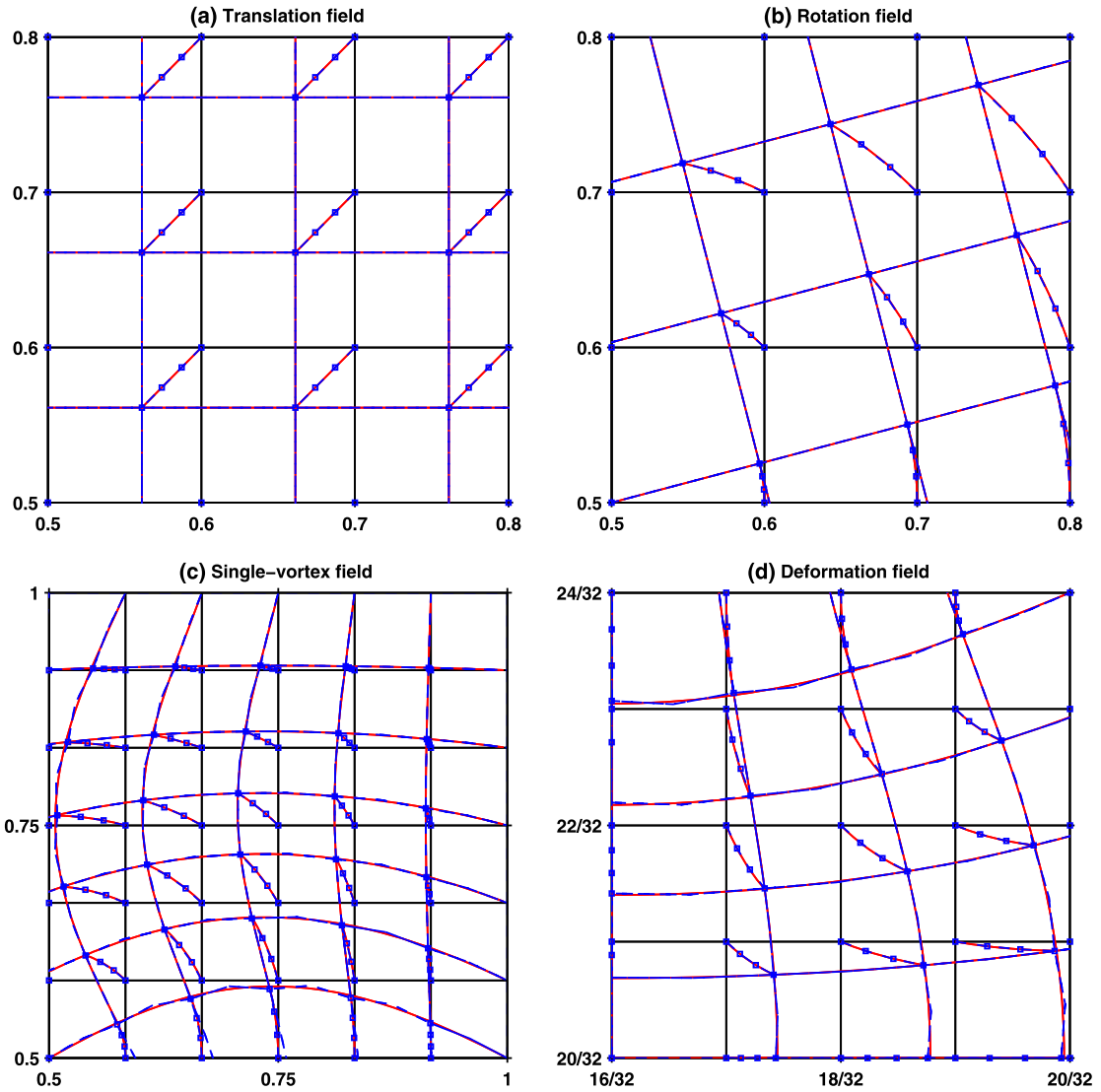


Fig. 9. The grid's pre-image and vertex's streaklines on representative unit-cells of the velocity field: (a) in simple translation, (b) in simple rotation, (c) in the single vortex in-a-box field, (d) in the 4-vortex deformation field. The solid red lines are the exact curves, while the dashed blue lines are calculated lines. Square markers are the vertices of polygonal chains representing the streaklines. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Table 1
Pre-image calculation errors E_1 computed for the entire grid, with $C_{max} = 1$. The convergence rates are in parentheses.

Grid size	8×8	16×16	32×32	64×64	128×128
Simple translation	2.16e−15	3.22e−15	1.72e−14	6.26e−14	2.58e−13
Simple rotation	8.68e−9	3.34e−10 (4.70)	1.39e−11 (4.59)	7.92e−13 (4.13)	1.54e−13
Single vortex in-a-box	7.00e−4	9.70e−5 (2.85)	1.39e−5 (2.81)	2.10e−6 (2.73)	3.10e−7 (2.75)
4-vortex deformation field	–	1.19e−3	2.72e−4 (2.13)	5.18e−5 (2.39)	8.18e−6 (2.67)

grids. The reconstructed DRs and pre-images do not overlap and are conforming to the edge of their neighbors (consistency of the streaklines). The pre-image calculation errors reported in Table 1 are computed for the entire domain, during one time-step, using the maximum Courant number $C_{max} = 1$.

In simple translation, the grid's pre-image is constructed exactly (to the machine precision), for all grid sizes. The increase in the value of E_1 with the mesh refinement is due to the accumulation of round-off errors. In the simple rotation,

Table 2
Geometrical errors and convergence rates in the single vortex deformation test cases.

	$E_g(32)$	$E_g(64)$	$O(32)$	$E_g(128)$	$O(64)$	$E_g(256)$	$O(128)$
$T = 0.5$							
Rider & Kothe [21]	7.29e−4	1.42e−4	2.36	3.90e−5	1.86	–	–
Stream [23]	5.51e−4	1.10e−4	2.32	3.38e−5	1.71	–	–
EMFPA [16]	4.45e−4	7.99e−5	2.48	2.04e−5	1.97	–	–
Mencinger & Zun [24]	4.68e−4	6.91e−5	2.76	2.07e−5	1.74	–	–
GPCA [25]	4.12e−4	7.32e−5	2.41	1.93e−5	1.93	–	–
CCU (proposed scheme)	3.20e−4	7.68e−5	2.06	1.32e−5	2.54	2.45e−6	2.43
$T = 2$							
Rider & Kothe [21]	2.36e−3	5.85e−4	2.01	1.31e−4	2.16	–	–
Stream [23]	2.37e−3	5.65e−4	2.07	1.32e−4	2.10	–	–
EMFPA [16]	2.14e−3	5.39e−4	1.99	1.29e−4	2.06	–	–
Mencinger & Zun [24]	2.11e−3	5.28e−4	2.00	1.28e−4	2.05	–	–
GPCA [25]	2.18e−3	5.32e−4	2.05	1.29e−4	2.03	–	–
CCU (proposed scheme)	1.86e−3	4.18e−4	2.15	9.62e−5	2.12	1.97e−5	2.29
$T = 8$							
Rider & Kothe [21]	4.78e−2	6.96e−3	2.78	1.44e−3	2.27	–	–
Stream [23]	3.72e−2	6.79e−3	2.45	1.18e−3	2.52	–	–
EMFPA [16]	3.77e−2	6.58e−3	2.52	1.07e−3	2.62	2.35e−4	2.19
Mencinger & Zun [24]	5.42e−2	7.85e−3	2.79	1.05e−3	2.90	–	–
GPCA [25]	–	–	–	1.17e−3	–	–	–
Owkes & Desjardins [46]	–	7.58e−3	–	1.88e−3	2.01	4.04e−4	2.22
CCU (proposed scheme)	3.81e−2	4.58e−3	3.06	1.00e−3	2.20	1.78e−4	2.59

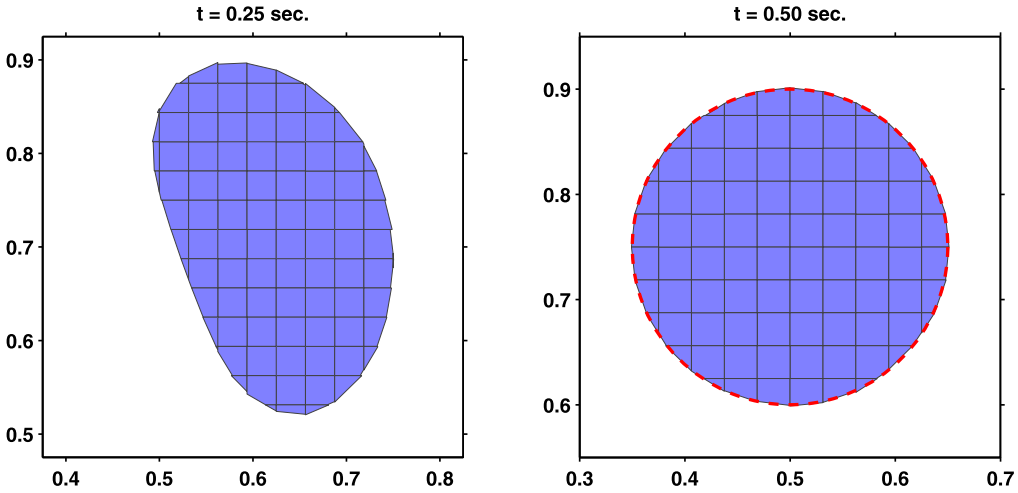


Fig. 10. Reconstructed interfaces in the single vortex deformation test with $T = 0.5$ and $C_{\max} = 1$, for the 32×32 grid. Snapshots at $t = T/2$ (left) and $t = T$ (right). The red dashed line shows the exact interface position at final time. (Mind the different scales.) (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

very small errors are generated. Careful examination of the calculated pre-image reveals that the pre-images of the grid vertices—traced backward—are determined exactly, because the rotational velocity field is linear in terms of space. The small errors come from the approximation of the streaklines as polygonal chains, while the exact streaklines are circle arcs. Errors in the area of the DR results in positioning errors of the mid-edge correcting vertex of the volume flux correction. The error E_1 is reduced with mesh refinements, with a convergence rate between 4 and 5, and it reaches the precision of accumulated round-off errors for the mesh 128×128.

In the single vortex in-a-box and the 4-vortex deformation fields, additional errors come from the backward tracing since the bicubic spatial interpolation of the velocity does not represent sine and cosine functions exactly. However, considering the coarseness of the mesh, these errors are considered relatively small. Bicubic splines have accuracy of order $O(\Delta x^4)$, but the number of cells increase as $O(1/\Delta x^2)$ with the mesh refinement, hence a formal second-order accuracy. However, the apparent convergence rate of the error is between 2 and 3.

In principle, the error in the simple rotation—only due to approximations of the streaklines with polygonal chains—would vanish if one uses a spline representation of the streaklines, as in the DRACS scheme [27,30]. However, the advection errors in the simple rotation tests, solely due to the streakline representation, are five orders of magnitude smaller than in the single vortex in-a-box tests, where backward tracing also contributes to the errors. The simple rotation tests also have

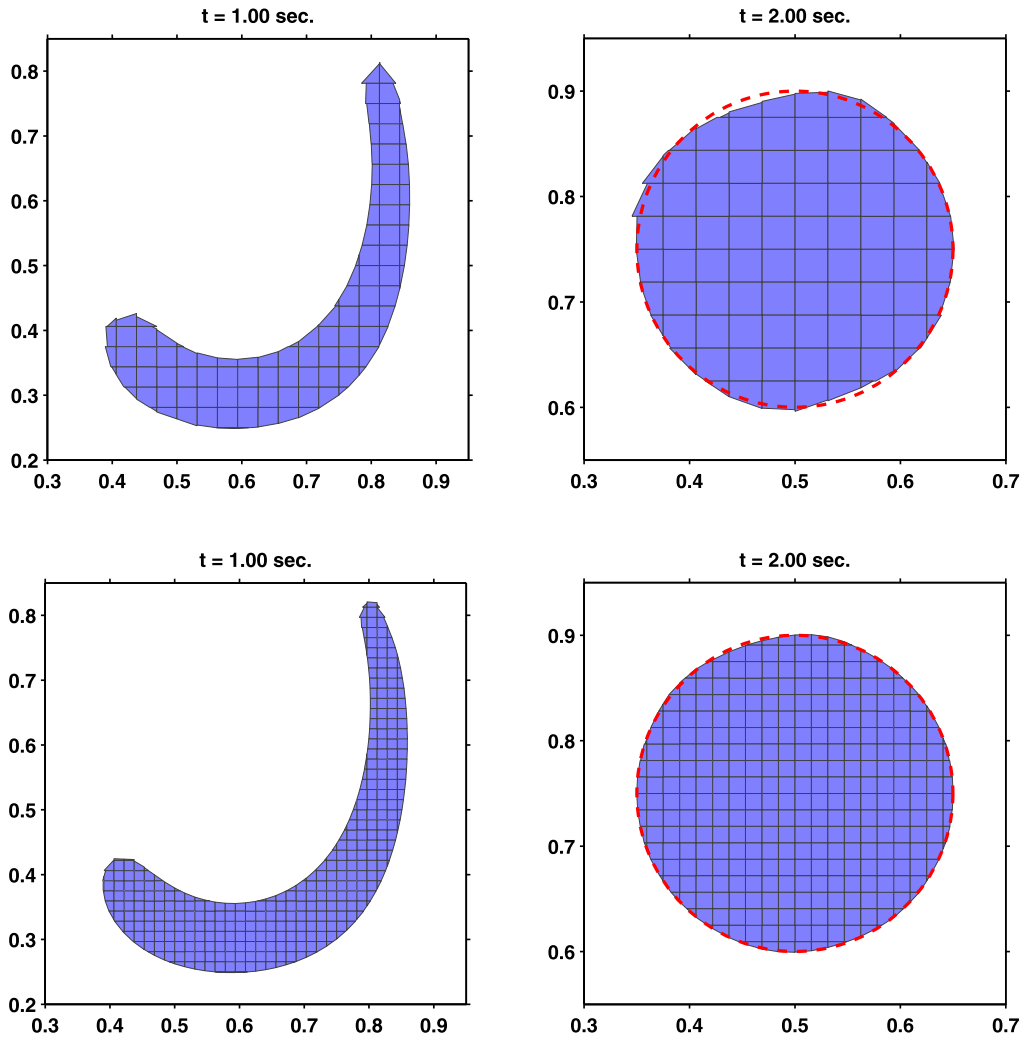


Fig. 11. Reconstructed interfaces in the single vortex deformation test with $T = 2$ and $C_{\max} = 1$, (top) for the 32×32 grid, and (bottom) for the 64×64 grid. Snapshots at $t = T/2$ (left) and $t = T$ (right). The red dashed line shows the exact interface position at final time. (Mind the different scales.) (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

higher convergence rates than the single vortex in-a-box tests. Therefore, the overall accuracy of advection is limited by the accuracy of the backward tracing, i.e. the Runge–Kutta method and the bicubic spatial interpolation, rather than the polygonal chain approximation.

3.2. Deformation in the single vortex test

In this section, the accuracy of the CCU advection scheme is tested in conjunction with the ELVIRA interface reconstruction algorithm. The deformation in a single vortex is a classical benchmark test, also used in [16,21,23–25]. The calculation domain is a rectangle $[0, 1] \times [0, 1]$. The liquid phase is initially a disk of radius 0.15 and centered at the position $(0.5, 0.75)$. The interface of the material deforms during the time interval $t = [0, T]$, by the single vortex in-a-box velocity field:

$$\psi(x, y, t) = -\frac{1}{\pi} \sin^2(\pi x) \sin^2(\pi y) \cos\left(\frac{\pi t}{T}\right). \tag{25}$$

The magnitude of the vortex is reversed at the time $t = T/2$; thus the deformed material theoretically recovers its initial shape at the final time T. The test is performed for three different vortex periods $T = 0.5, 2, 8$, with different grid spacing of the Cartesian grids $h = 1/32,\ 1/64,\ 1/128$ and $1/256$. All the simulations are performed with $C_{\max} = 1$.

The CCU advection scheme rigorously conserves the liquid volume to the machine precision. Geometrical errors are measured as the 1-norm of the difference between the initial and final liquid volume fraction inside all the cells:

$$E_g(n) = \sum_{i,j}^{n} A_{i,j} \left| f_{i,j}^{T} - f_{i,j}^{0} \right|, \tag{26}$$

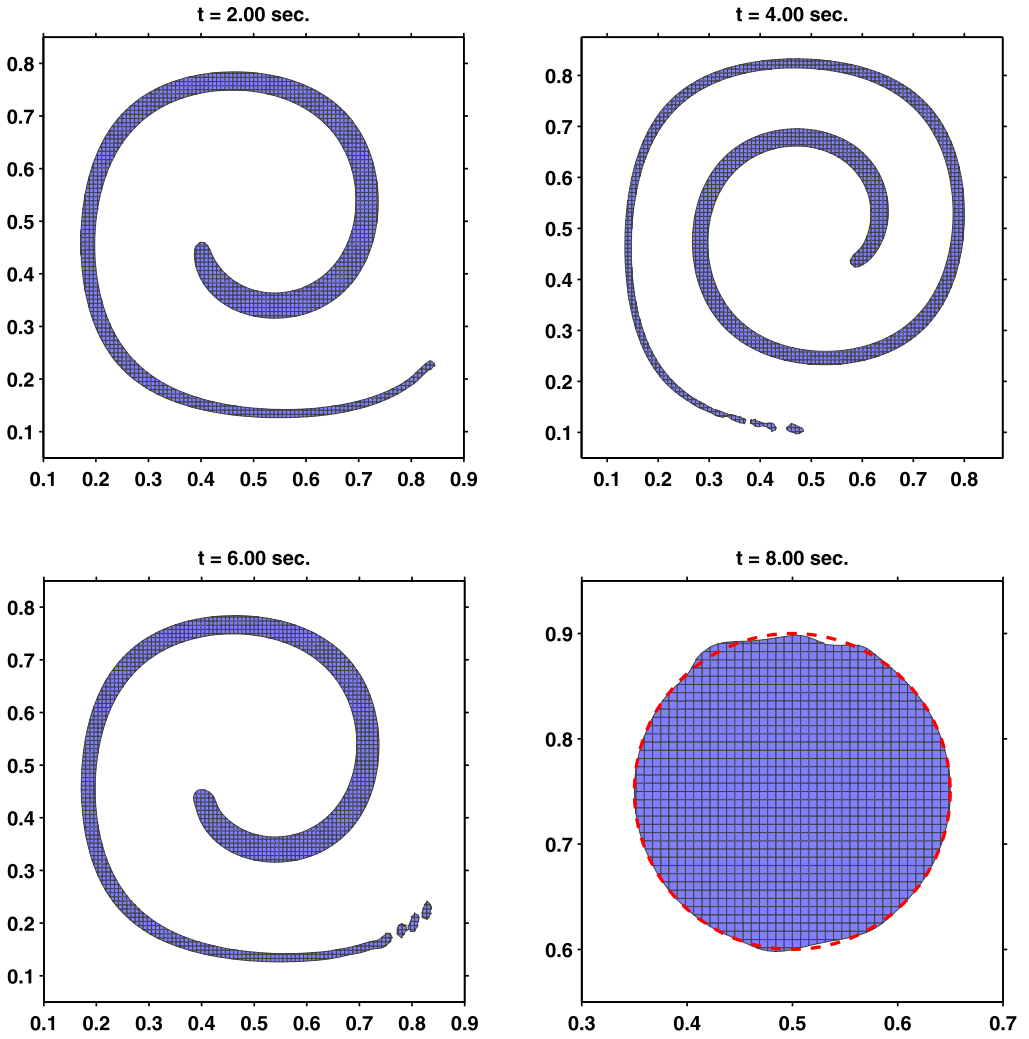


Fig. 12. Reconstructed interfaces in the single vortex deformation test with $T = 8$ and $C_{\max} = 1$, for the 128×128 grid. Snapshots at $t = T/4$ (top-left), $t = T/2$ (top-right), $t = 3T/4$ (bottom-left) and $t = T$ (bottom-right). The red dashed line shows the exact interface position at final time. (Mind the different scales.) (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

where $A_{i,j}$ is the area of the i, jth grid cell, and n is the number of grids along each dimension. The convergence rate of the error with mesh refinement is given by:

$$O(n) = \frac{\ln(E_g(2n)/E_g(n))}{\ln(1/2)}. \tag{27}$$

The geometrical errors obtained in our simulation are shown in Table 2 and compared with results from other advection schemes. The geometrical error decreases with mesh refinements at a rate between 2 and 3. In most of the cases, the CCU scheme performs better than the best of the other schemes, with 15–30% less geometrical errors. In the particular case where $T = 0.5$, the CCU scheme is outperformed by the GPCA and the Mencinger and Zun schemes, on the 64×64 grid. However, this could be due to the difference in the interface reconstruction algorithm. Formally, the LVIRA algorithm used in [16,21,23–25] is more accurate than the ELVIRA algorithm we use, although both methods return similar results in most cases.

Figs. 10–13 show the reconstructed interfaces at various times of the simulation, for different vortex periods T and grid sizes. The theoretical position of the interface at the end of the simulation is represented by the red dashed line. In simulations with the vortex period $T = 8$, the deformed material is extensively stretched. In theory, the material has the same shape at the times $t = T/4$ and $t = 3T/4$. In Fig. 12, a visual comparison of the geometry at the times $t = T/4$ and $t = 3T/4$, shows that geometrical errors arise from the head and the tail of the deformed material. The same observation is also made when comparing the reconstructed interfaces between the different grid sizes. These regions of high geometrical errors are located where the interface has a high curvature, or forms thin filament below the thickness of 2 grid cells. In these regions, the interface reconstruction algorithm fails in capturing the interface accurately. The observed filament

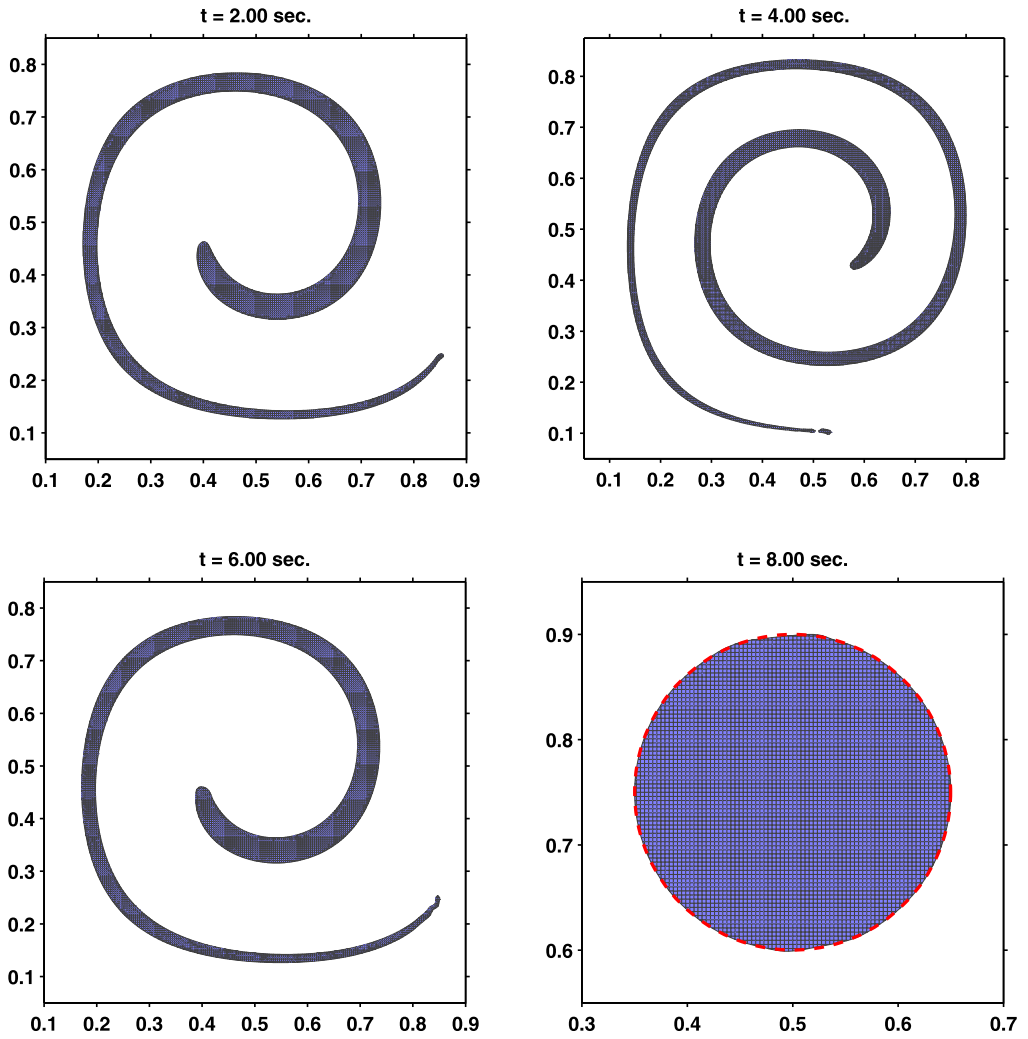


Fig. 13. Reconstructed interfaces in the single vortex deformation test with $T = 8$ and $C_{\max} = 1$, for the 256×256 grid. Snapshots at $t = T/4$ (top-left), $t = T/2$ (top-right), $t = 3T/4$ (bottom-left) and $t = T$ (bottom-right). The red dashed line shows the exact interface position at final time. (Mind the different scales.) (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

break-up is a numerical artifact, which should not be confused with actual fluid thread break-up due to minimization of the surface tension energy. This is not only a drawback of the LVIRA and the ELVIRA algorithm, but the PLIC representation in general. Therefore, the simulation results in the cases $T = 8$, for the coarse grids 32×32 and 64×64, are not very relevant to assess the accuracy of the CCU scheme, as the large geometrical errors come from the PLIC representation rather than the advection.

3.3. Deformation in the 4-vortex test

The 4-vertex deformation field is another benchmark test for the surface-tracking methods. Traditionally, it is considered to be a very difficult test case to solve with the VOF method, because of the appearance of very thin filaments of material that are excessively stretched. The liquid phase initially forms a disk of radius 0.15, located at the center of the $[0, 1] \times [0, 1]$ domain. The velocities in the 4-vortex deformation field are derived from:

$$\psi(x, y, t) = \frac{1}{4\pi} \sin\big(4\pi\,(x + 0.5)\big) \cos\big(4\pi\,(y + 0.5)\big) \cos\left(\frac{\pi t}{T}\right). \tag{28}$$

As in the previous benchmark test, the amplitude of the deformations is reversed at the time $t = T/2$, and the material theoretically recovers its initial shape at the end of the simulation. The test is performed on various grid sizes, for the deformation periods $T = 1$ and $T = 2$. Two maximum Courant numbers $C_{\max} = 1$ and $C_{\max} = 0.1$ are used in the simulations.

The reconstructed shapes of the deformed material with the ELVIRA method are depicted in Figs. 14 and 15, where $T = 1$ and $T = 2$, respectively. For $T = 1$, the PLIC is able to represent the formation of the filament, on the intermediate grid

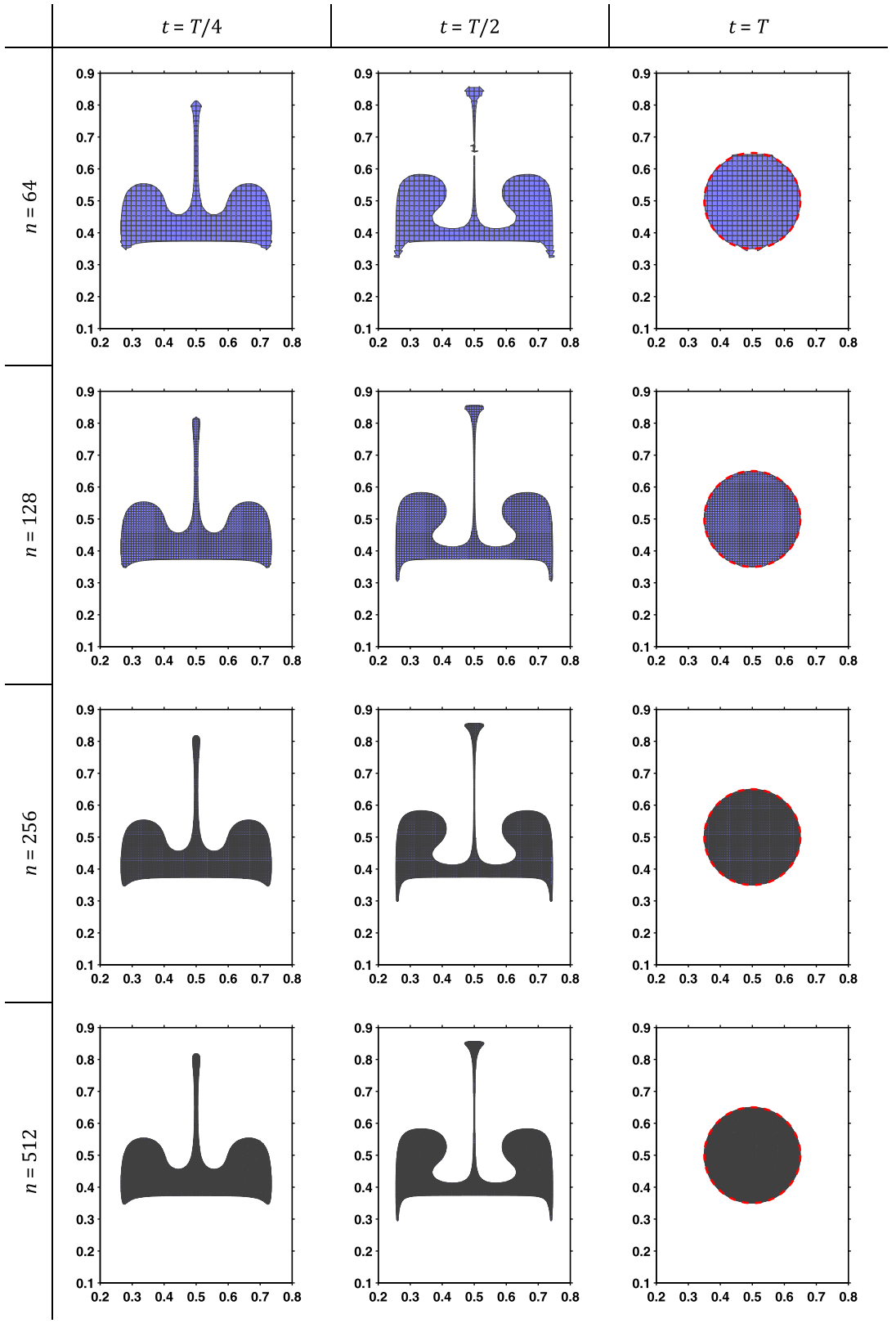


Fig. 14. Snapshots of the reconstructed interface in the 4-vortex deformation test with $T = 1$ and $C_{\max} = 1$.

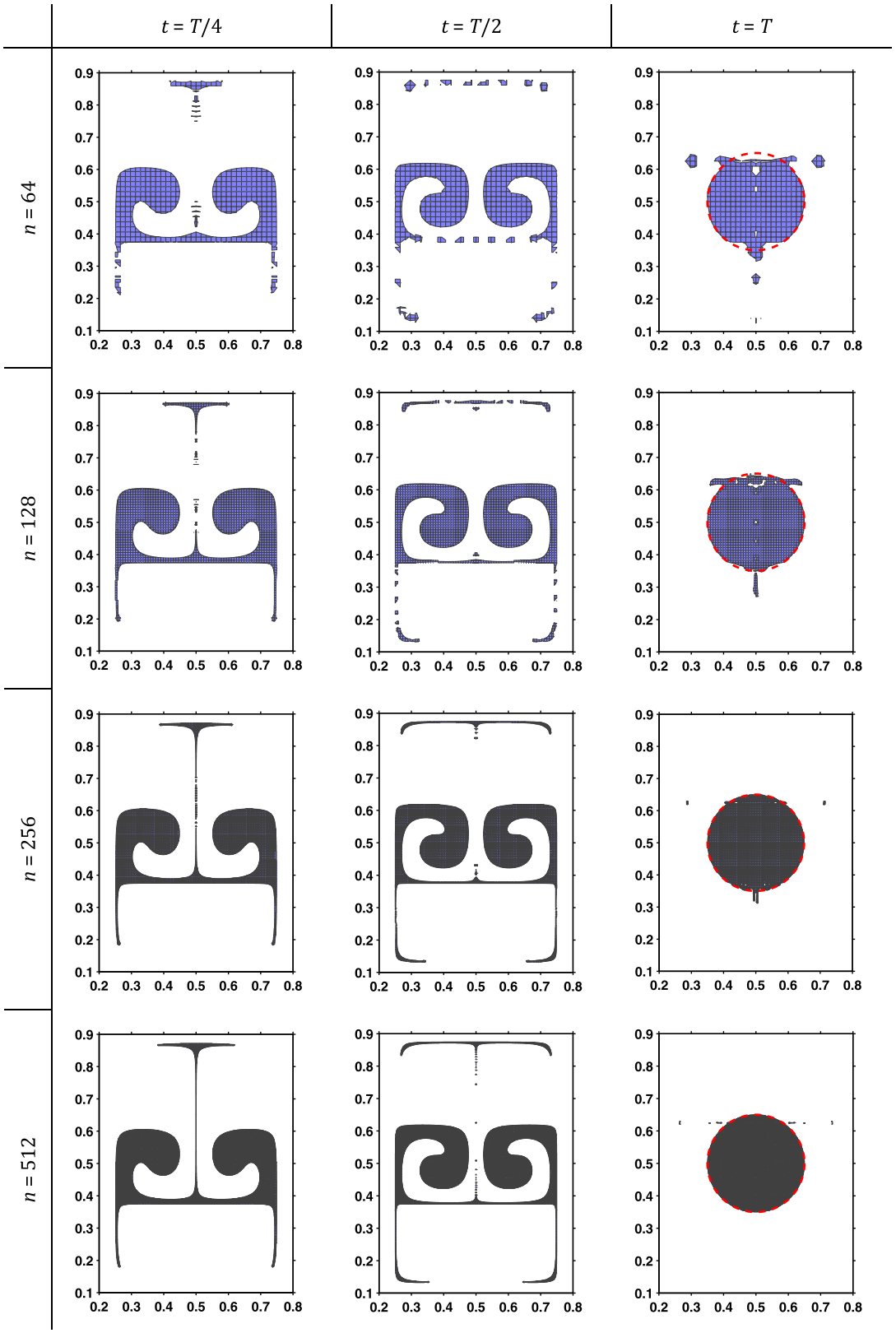


Fig. 15. Snapshots of the reconstructed interface in the 4-vortex deformation test with $T = 2$ and $C_{\max} = 1$.

Table 3
Geometrical errors and convergence rates in the 4-vertex deformation test cases.

	$E_g(64)$	$E_g(128)$	$O(64)$	$E_g(256)$	$O(128)$	$E_g(512)$	$O(256)$
$T=1$							
Rider & Kothe [21]	1.69e−3	4.36e−4	1.95	–	–	–	–
CCU $C_{max}=1$	1.04e−3	2.78e−4	1.90	8.02e−5	1.80	2.13e−5	1.92
CCU $C_{max}=0.1$	3.35e−3	4.75e−4	2.82	1.15e−4	2.04	3.12e−5	1.89
$T=2$							
Rider & Kothe [21]	1.12e−2	5.95e−3	0.91	–	–	–	–
CCU $C_{max}=1$	1.07e−2	5.58e−3	0.94	1.47e−3	1.93	3.69e−4	1.99
CCU $C_{max}=0.1$	1.42e−2	7.31e−3	0.96	1.60e−3	2.19	4.42e−4	1.86

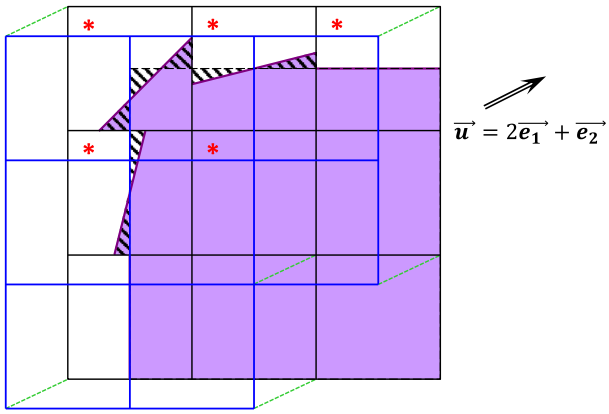


Fig. 16. Advection errors in simple translation of a rectangular corner. The reconstructed shape of the material is colored in purple; hatched areas represent static reconstruction errors. The grid cells marked with a red asterisk are subjected to advection errors, in spite of the exact pre-image calculation in blue. Note that only the cells near the interface have their liquid volume fraction updated. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

128×128. However, For $T=2$, the material experiences very large deformations, and the PLIC has difficulties representing the elongated thin filaments, even on the finest 512×512 grid. The geometrical errors and their convergence rates are reported in Table 3. Our results are compared with the available data of Rider and Kothe [21] for the coarse grids. This benchmark test triggers the occurrence of self-intersecting DRs, as depicted in Fig. 6(b). Self-intersecting DRs are only handled correctly by schemes producing edge-matching fluxes (which is not the case of the Rider & Kothe's scheme). In the case $T=1$, the CCU scheme reduces the geometrical errors by one third, as compared to the Rider and Kothe's scheme. However, in the case $T=2$, the CCU scheme produce only slight improvements. Relatively large geometrical errors arise from the filament break-up due to under-resolution of the mesh. The limiting factor for the accuracy is the improper PLIC interface representation. The convergence rate of the geometrical error is around 2, which is consistent with the second-order accuracy of the ELVIRA algorithm [13].

Paradoxically, the geometrical error increases when the Courant number decreases, which is inconsistent with the temporal accuracy of advection schemes. This is also due to errors in the interface reconstruction. Indeed, the accuracy of the interface reconstruction algorithm is independent on the time-step increment, but interface reconstruction errors accumulate as the number of time-step increases.

4. Discussions

4.1. Interface reconstruction errors

Overall, the CCU algorithm compares favorably to the other 2D unsplit advection schemes using the PLIC interface representation. However, geometrical errors persist mainly because of errors in the interface representation. An analysis of interface reconstruction errors is reported in [33]. As also shown in [27], poor interface representations (seen by substantial discontinuities of the interface) generate advection errors. Fig. 16 shows an example where advection errors are produced because of static interface reconstruction errors, in spite of an exact pre-images calculation. The use of more advanced interface representation techniques would be necessary in order to overcome this type of error.

Quadratic interface reconstruction has been used in [17,34–36] to improve the representation of regions with high curvature. However, intersections of quadratic surfaces are more difficult to compute. Another technique, the improved PLIC-VOF method of López et al. [37], successfully achieves the accurate representation of thin filaments with two non-secant line segments whose positions are tracked by markers. Both techniques are improvements of the interface representation and

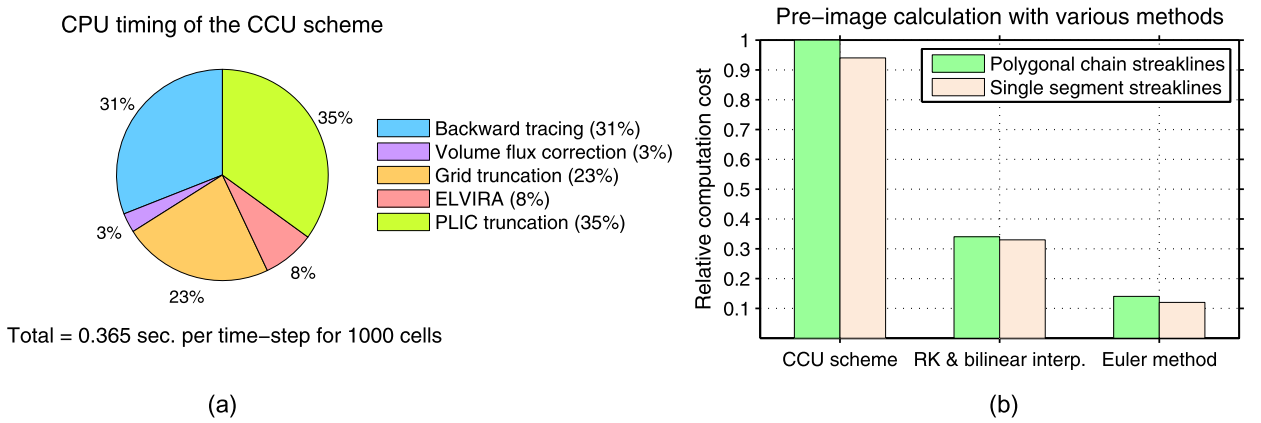


Fig. 17. Timing data: (a) distribution of the total calculation time between the different steps of the VOF solver, (b) comparison of the computation costs, for the pre-image calculation, with downgraded methods from the original CCU scheme.

produce better results than the PLIC in the single vortex benchmark test. Other remaining issues of the PLIC-VOF method are (i) the improper representation of sharp corners, (ii) the interface reconstruction of multi-materials in cells containing tri-points, and (iii) the accurate tracking of sub-grid particles of fluid, such as small droplets or air bubbles.

Some of these issues are handled better with other types of interface tracking methods, notably the hybrid markers-VOF method of Aulisa [38,39] and the PAM method of Zhang and Liu [28,29]. Both methods perform better than the PLIC-VOF when representing the interface, due to their use of Lagrangian geometrical objects such as markers or polygons. These two methods also have the ability to represent geometrical shapes below the grid resolution (bubbles or droplets). However, these improvements come at the price of tedious calculations. For instance, the hybrid marker-VOF method requires dynamical redistribution of the markers under the constraint of volume conservation, in order to keep a uniform marker density along the interface. The PAM method uses readjustment procedures, to shrink and expand the volume of polygons, and to reduce their number of vertices. Improvements of these readjustment procedures have been presented in [29]. In addition, both the original and the improved PAM methods require special algorithms to successfully merge interfaces.

On the contrary, Eulerian *surface capturing* techniques are relatively easier to implement and they handle interface merging automatically. Finally, the mentioned drawbacks of the PLIC-VOF method are rather due to the PLIC representation of the interface than the VOF approach itself. In the authors' opinion, the VOF method could reach equivalent accuracy as the Lagrangian methods, if more precise techniques for the interface representation were developed. In particular, we suggest the use of piecewise continuous bilinear interfaces, i.e. two-segment polygonal chains, to improve the representation of the sharp corners and the high curvatures.

4.2. Computation cost

This section provides timing data of the CCU advection scheme. Timing data are interesting when assessing the efficiency of an algorithm; however, direct comparisons with other values from the literature are not relevant, since the computational cost depends on the hardware, the programming language, and the coding style of the implementation. In this work, the CCU advection scheme has been implemented in Matlab®. The code was vectorized with the use of array operations, in order to optimize computational efficiency. However, since Matlab® is an interpreted language, its speed of calculation is slower than compiled codes converted in executable machine codes. Our code ran on a workstation with a dual hexa-core microprocessor T7500 dual Hex X5690 @ 3.46, 96 GB DDR3 1333 MHz.

The calculation time of the CCU algorithm was recorded in the single vortex in-a-box benchmark test, with $T = 2$ and $C_{\max} = 1$, for the 256×256 grid. In this test case, the number of grid cells that undergo an update of their liquid volume fraction (hereinafter referred to as the active cells) varies between 924 cells at the initial time and 1814 cells in the most stretched configuration at the time $t = T/2$. On the one hand, the computational cost for the pre-image calculation (backward tracing and volume flux correction) is directly proportional to the number of active cells. On the other hand, the computational costs for the interface reconstruction and the calculation of the intersections of the pre-images with the PLIC are proportional to the number of interfacial cells. In order to have a coherent scaling of the calculation time, we have chosen a test case without formation of thin filaments, such that the ratio of interfacial cells to the total number of active cells stays approximately to the proportion 1/3. The solving of the test case required 336 time-steps in total. The calculation time was recorded for each time steps and scaled to 1000 active cells. The median calculation time, for the entire VOF solver, was 0.365 seconds per time-step for 1000 active cells. Fig. 17(a) shows the distribution of the total computational cost between the different steps of the VOF solver. The pre-image calculation, which is the originality of the CCU scheme, represents about one third of the whole calculation time. Remaining computational costs are shared between the ELVIRA interface reconstruction algorithm, and the truncation of the pre-image polygons with the grid lines and the PLIC.

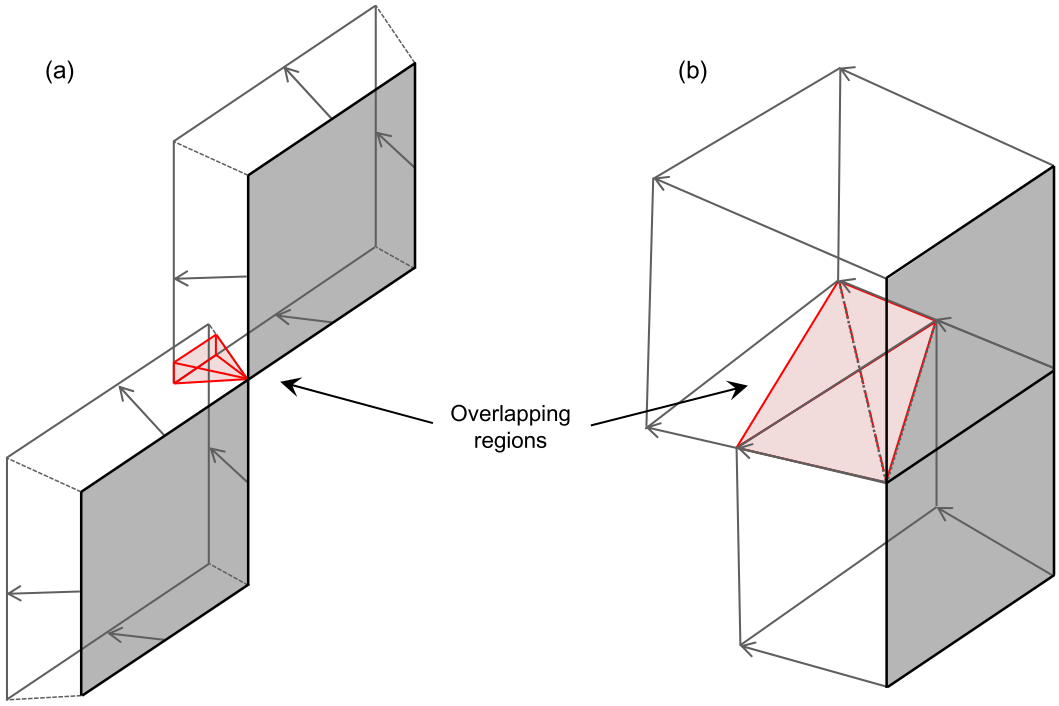


Fig. 18. Overlapping regions of non-conforming DR: (a) for the FMFPA scheme [43], (b) for the voFoam scheme [44].

Additional timing tests were performed with simpler version of the CCU scheme, downgraded with (i) bilinear interpolations instead of bicubic splines, (ii) the Euler method—as in the EMFPA scheme—instead of the Runge–Kutta method, and (iii) the representation of streaklines with single segments instead of polygonal chains. These tests provide an estimation of the additional computational cost associated to the CCU scheme. The computational cost of the pre-image calculation is displayed for the different schemes in Fig. 17(b), where the original CCU scheme is scaled to 1. The data shows that the bilinear interpolation reduces the computational cost of the pre-image calculation by 2/3. Moreover, the use of the Euler method, which does not require any spatial interpolation, further decreases the computational cost by 3/5. Nevertheless, when these timing data of the pre-image calculation are related to the entire VOF solver, the additional computational cost associated to the CCU scheme does not exceed 20% or 30% of the total cost, as compared to the advection schemes with bilinear spatial interpolations and the Euler method, respectively. It is interesting to note that the calculation of the streaklines as polygonal chains does not add a significant computational cost, as compared to the calculation of single segment streaklines. This is due to the fact that the backward tracing of the intermediate vertices 1' and 1″ of the polygonal chain does not require calculation of additional spatial interpolation functions, since it uses the same as for the backward tracing of the end-point A of the streakline.

Finally, one should keep in mind that the reported calculation time of 0.365 seconds per time-step for 1000 active cells is an illustration of the computation cost, for one possible implementation of the CCU scheme in Matlab®. For instance, the total calculation time will be significantly reduced, if the scheme is programmed in a low-level compiled language, such as FORTRAN or C/C++. In addition, we have not taken advantage of parallel computing, despite the explicit nature of the VOF method and the CCU scheme, which inherently makes them easily parallelizable.

4.3. Extension to 3D

The extension of 2D unsplit VOF-advection schemes into 3D is often cited as a difficult and tedious task. To date, 3D directional-split schemes [19,20,40] are more popular despite being less accurate. Nevertheless, several implementations of 3D unsplit schemes have been reported in the literature. Rider and Kothe, and Puckett's schemes have been extended to 3D in [41] and [42], respectively. Two 3D variants of EMFPA have also been proposed: the FMFPA [43] and voFoam [44] schemes. The two schemes differ from each other in how they construct the DRs. FMFPA constructs face-matched flux polyhedra using edge-centered velocities. The geometrical toolbox [45] has specially been developed to facilitate its implementation. FMFPA produces adjacent DRs with matched coplanar faces, yet overlapping with their diagonal neighbors sharing one corner, as depicted in Fig. 18(a). Alternatively, voFoam uses the grid's vertex velocities to produce truly edge-matched fluxes. Consequently, each face of the DR is defined by 4 vertices that are not coplanar in the general case, see Fig. 19(a). The resulting non-planar surfaces need to be triangulated into planar faces. In addition, the voFoam scheme produces non-conforming

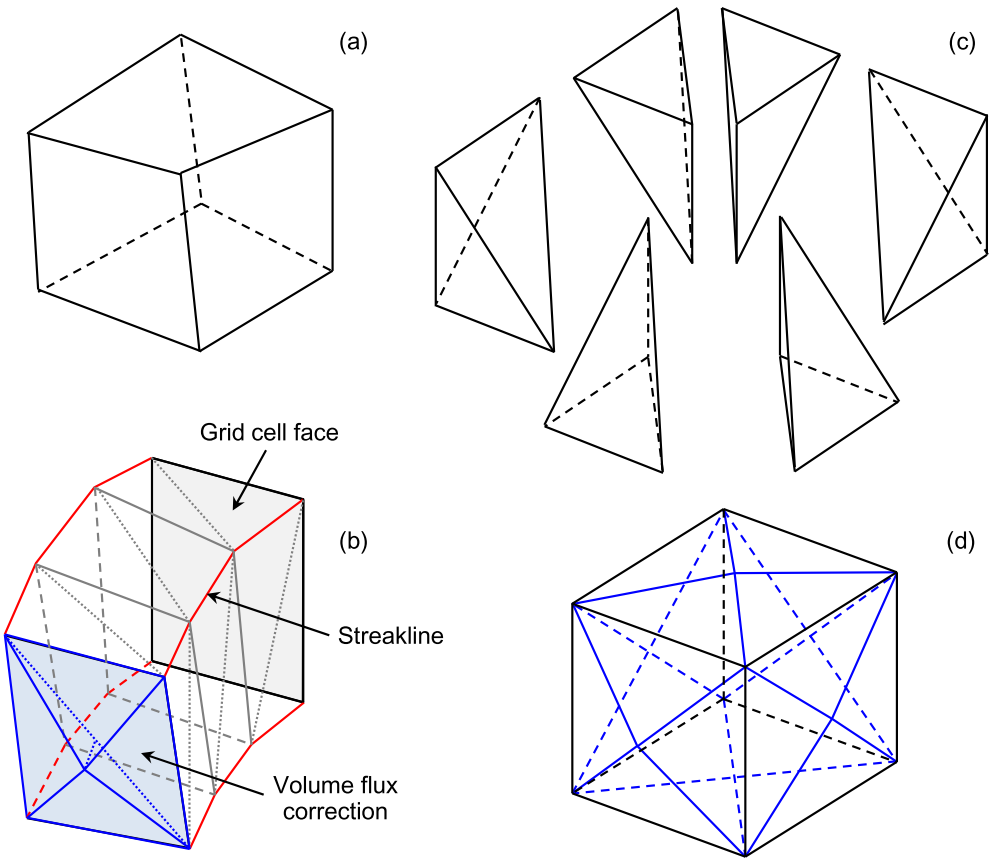


Fig. 19. (a) Quadrilaterally-faced hexahedron with non-planar ruled surfaces. (b) DR calculation delimited with polyhedral terrains, and pyramidal volume flux correction. (c) Exploded view of the volume decomposition of a hexahedron into 6 simplexes. (d) Final pre-image polyhedron with 14 vertices and 24 faces.

DRs (similarly as EMFPA), therefore the triangulations of adjacent DR surfaces do not match, forming overlapping regions or gaps, as depicted in Fig. 18(b). As a result, none of FMFPA and voFoam is conservative.

Frameworks for 3D conservative and bounded unsplit schemes have only been described in very recent publications [46,47]. The conforming of adjacent DRs—which is a *sufficient condition* for conservativeness in 2D—is here a *necessary condition* for conservativeness in 3D. In this regard, the CCU scheme, which produces edge-matched conforming DRs, is a good candidate for 3D conservative unsplit advection. The backward tracing of the grid's vertices in 3D remains essentially identical as in the 2D case. The bicubic spatial interpolation is replaced by a tricubic interpolation [48]. The DR calculation in 3D still uses polygonal chains to approximate the streaklines of the grid's vertices. The resulting DR polyhedron is bounded by polyhedral terrains and it is composed of three adjoining hexahedra, as represented in Fig. 19(b). As shown in [46,49], the surface triangulation of each quadrilaterally-faced hexahedron having non-planar surfaces is equivalent to volume decomposition into six simplexes; see Fig. 19(c). The simplex decomposition facilitates the calculation of the polyhedral volumes. Without loss of generality, the volume of an arbitrary simplex defined by its vertex's positions $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$ is given by:

$$V^{\text{simplex}} = \frac{1}{6}\left|(\mathbf{a}-\mathbf{d})\cdot\left[(\mathbf{b}-\mathbf{d})\times(\mathbf{c}-\mathbf{d})\right]\right|. \tag{29}$$

The simplex decomposition is used to compute the volume of the DR. Note that DR polyhedrons may have self-intersections. The different scenarios of self-intersection must be checked, and non-admissible self-intersections (not involving the cell face) must be regularized, similarly as in the 2D case. Checking all the potential intersections of the polygonal terrains bounding the DR might be a tedious task. However, the occurrence of the self-intersections can be detected with the use of the sign convention, introduced in [46], relative to the decomposition of polyhedra into simplex volumes. Thus, the regularization of non-admissible DRs, and the calculation of the signed volume of self-intersecting DRs, can be performed with only few computations of plane–plane intersections. The full details about the sign convention for the decomposition of polyhedra into simplex volumes are found in [46]. Finally, the 3D volume flux corrections are applied as pyramidal volumes, producing non-overlapping and conforming DRs. As a result, the final pre-image polyhedron has 14 vertices and 24 faces, see Fig. 19(d), which is decomposed into 20 simplexes, similarly as in [46].

The two unsplit schemes [46] and [47], for hexahedral and tetrahedral meshes respectively, also achieve conservativeness and boundedness in 3D by means of similar volume flux corrections as in the CCU scheme. Nevertheless, the originalities that differentiate the CCU method from the two former conservative unsplit schemes are:

- the tricubic interpolations of the velocity field (instead of trilinear), for more accurate higher-order backward tracing,
- the more precise representation of the DR polyhedron, bounded by polygonal terrains whose edges are polygonal chains approximating the streaklines of the grid's vertices,
- the cellwise approach of the VOF update.

The accuracy gained in both backward tracing and DR representation decreases the magnitude of volume correction needed, thus reducing pre-image calculation errors.

For the sake of clarity, the CCU advection scheme has here been presented assuming the case of a Cartesian grid. However, there is no restriction on the mesh morphology. Unstructured quadrilateral/hexahedral meshes can be used without any differences. In the case of unstructured triangular/tetrahedral meshes, minor modifications are required. The bi/tri-cubic interpolations are replaced by bi/tri-quadratic interpolations, dropping the continuity of cross-derivative terms. Additionally, the pre-image of a tetrahedral element is a polyhedron with 8 vertices and 12 faces, which is efficiently decomposed into 12 simplexes.

Finally, several planar interface reconstruction algorithms have been developed for 3D problems. The least-square fit method (LVIRA) has been extended to 3D Cartesian grids in [42]. The correct positioning of the planar interfaces inside rectangular cells is obtained from the analytical relations given in [50]. Other reconstruction techniques [51] have been implemented to determine smooth second-order accurate planar interface in 3D tetrahedral unstructured meshes. Useful analytic relations for the correct positioning of planar interfaces inside arbitrary triangular and tetrahedral cells are also provided in [52].

5. Conclusion

We have presented a new cellwise conservative unsplit advection scheme for the volume-of-fluid method. The CCU scheme is based on the backward tracing of the cells' pre-images, using a fourth-order Runge–Kutta method with bicubic spatial interpolation of the velocity field. The donating regions of the fluxes are calculated as polygons, whose edges are polygonal chains representing the streaklines of the cell's vertices. The volume conservation is enforced through a robust volume flux correction algorithm, producing non-overlapping and conforming adjacent donating regions. As a result, the CCU scheme is conservative and bounded. In addition, the cellwise update of the liquid volume fraction does not require the computation of the liquid volume inside the fluxes' donating regions since the liquid volume is solely evaluated in the cells' pre-images. Consequently, it requires fewer computations of polygonal intersections. Finally, the CCU advection is not subjected to restrictions on the Courant number.

Numerical simulations show that the CCU scheme overall performs better than the other PLIC-VOF methods in the classical benchmark tests. The accuracy of the CCU scheme is gained from the bicubic spatial interpolation and the representation of streaklines with polygonal chains. A measure of the pre-image calculation error has been proposed to provide an upper bound for its contribution to the global advection errors, independently of interface reconstructions. The convergence rates of the pre-image calculation errors with mesh refinements are between 2 and 3 in complex flows.

The contribution of the interface reconstruction to the global advection error has also been discussed since some of the geometrical errors measured in the benchmark tests are due to the inability of the reconstructed planar interfaces to properly represent high curvatures and thin material filaments. Therefore, we suggest future research efforts on the improvement of the VOF method to be directed toward the development of more precise interface representation techniques, transcending the drawbacks of the PLIC.

We have shown that the additional computational cost of the CCU scheme is mainly due to the calculation of the bicubic spatial interpolations, while the streakline representation with polygonal chains has a negligible impact on the total calculation time. Moreover, the overall computational cost is dominated, in our implementation, by the rest of the VOF algorithm, i.e. the interface reconstruction and the calculation of polygonal intersections.

Finally, we have discussed the potential extension of the CCU advection to 3D calculations. It appears that the CCU scheme is a good candidate to achieve accurate, conservative and bounded advection in 3D, thanks to the robust calculation of conforming and non-overlapping pre-images and donating regions. Furthermore, the CCU scheme can be adapted to triangular and tetrahedral unstructured meshes, with only minor modifications.

Acknowledgements

The authors would like to acknowledge the support of the Scientific Research Council on Technology and Production Sciences (FTP) (Contract No. 09-072888, OPTIMAC), which is part of the Danish Council for Independent Research (DFF). The second author was funded by the FTP Contract No. 4005-00381, Decreasing concrete-related CO_2 emissions through optimization of rheology.

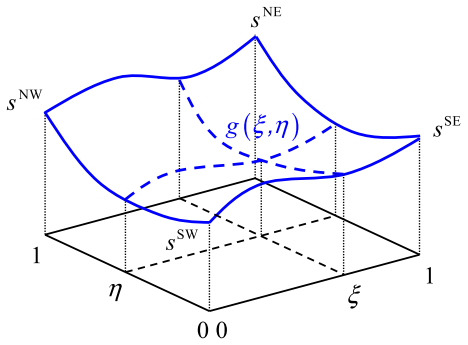


Fig. 20. Bicubic spline interpolation inside a rectangular grid cell, using normalized local coordinate system.

Appendix A. Bicubic spline interpolation

The bicubic spline interpolation g of a scalar field s (either one of the velocity components u, v, or the streamfunction ψ) is a polynomial function with 16 terms:

$$g(\xi,\eta)=\sum_{i=0}^{3}\sum_{j=0}^{3}a_{ij}\xi^i\eta^j=a_{00}+a_{01}\eta+a_{02}\eta^2+a_{03}\eta^3+a_{10}\xi+a_{11}\xi\eta+a_{12}\xi\eta^2+a_{13}\xi\eta^3+\cdots$$
$$+a_{20}\xi^2+a_{21}\xi^2\eta+a_{22}\xi^2\eta^2+a_{23}\xi^2\eta^3+a_{30}\xi^3+a_{31}\xi^3\eta+a_{32}\xi^3\eta^2+a_{33}\xi^3\eta^3, \tag{30}$$

where $(\xi,\eta)\in[0,1]^2$ are the normalized local coordinates inside the rectangular grid cell, see Fig. 20. The value of the field s is known at the cell's vertices $(0,0)$, $(1,0)$, $(1,0)$, $(1,1)$. The first and cross derivatives s_x, s_y and s_{xy} are further evaluated at these vertices by finite-differences. Fitting these 4 values at the 4 corners provides 16 equations for the coefficients a_{ij}:

$$\begin{aligned}
s^{SW}&=g(0,0)=a_{00},\\
s^{SE}&=g(1,0)=a_{00}+a_{10}+a_{20}+a_{30},\\
s^{NW}&=g(0,1)=a_{00}+a_{01}+a_{02}+a_{03},\\
s^{NE}&=g(1,1)=a_{00}+a_{01}+a_{02}+a_{03}+a_{10}+a_{11}+a_{12}+a_{13}+\cdots\\
&\quad+a_{20}+a_{21}+a_{22}+a_{23}+a_{30}+a_{31}+a_{32}+a_{33},
\end{aligned} \tag{31}$$

$$\begin{aligned}
\Delta x s_x^{SW}&=\frac{\partial g}{\partial\xi}(0,0)=a_{10},\\
\Delta x s_x^{SE}&=\frac{\partial g}{\partial\xi}(1,0)=a_{10}+2a_{20}+3a_{30},\\
\Delta x s_x^{NW}&=\frac{\partial g}{\partial\xi}(0,1)=a_{10}+a_{11}+a_{12}+a_{13},\\
\Delta x s_x^{NE}&=\frac{\partial g}{\partial\xi}(1,1)=a_{10}+a_{11}+a_{12}+a_{13}+2a_{20}+2a_{21}+2a_{22}+\cdots\\
&\quad+2a_{23}+3a_{30}+3a_{31}+3a_{32}+3a_{33},
\end{aligned} \tag{32}$$

$$\begin{aligned}
\Delta y s_y^{SW}&=\frac{\partial g}{\partial\eta}(0,0)=a_{01},\\
\Delta y s_y^{SE}&=\frac{\partial g}{\partial\eta}(1,0)=a_{01}+a_{11}+a_{21}+a_{31},\\
\Delta y s_y^{NW}&=\frac{\partial g}{\partial\eta}(0,1)=a_{01}+2a_{02}+3a_{03},\\
\Delta y s_y^{NE}&=\frac{\partial g}{\partial\eta}(1,1)=a_{01}+2a_{02}+3a_{03}+a_{11}+2a_{12}+3a_{13}+\cdots\\
&\quad+a_{21}+2a_{22}+3a_{23}+a_{31}+2a_{32}+3a_{33},
\end{aligned} \tag{33}$$

$$\Delta x\Delta y s_{xy}^{\mathrm{SW}} = \frac{\partial^2 g}{\partial\xi\,\partial\eta}(0,0) = a_{11},$$
$$\Delta x\Delta y s_{xy}^{\mathrm{SE}} = \frac{\partial^2 g}{\partial\xi\,\partial\eta}(1,0) = a_{11} + 2a_{21} + 3a_{31},$$
$$\Delta x\Delta y s_{xy}^{\mathrm{NW}} = \frac{\partial^2 g}{\partial\xi\,\partial\eta}(0,1) = a_{11} + 2a_{12} + 3a_{13},$$
$$\Delta x\Delta y s_{xy}^{\mathrm{NE}} = \frac{\partial^2 g}{\partial\xi\,\partial\eta}(1,1) = a_{11} + 2a_{12} + 3a_{13} + 2a_{21} + 4a_{22} + 6a_{23} + 3a_{31} + 6a_{32} + 9a_{33}. \tag{34}$$

The system of linear relations (31)–(34) can be expressed in the matrix form

$$\boldsymbol{A}\alpha = x, \tag{35}$$

where

$$\alpha = [\, a_{00} \;\; a_{10} \;\; a_{20} \;\; a_{30} \;\; a_{01} \;\; a_{11} \;\; a_{21} \;\; a_{31} \;\; a_{02} \;\; a_{12} \;\; a_{22} \;\; a_{32} \;\; a_{03} \;\; a_{13} \;\; a_{23} \;\; a_{33} \,]^{\mathrm{T}},$$
$$x = \big[\, s^{\mathrm{SW}} \;\; s^{\mathrm{SE}} \;\; s^{\mathrm{NW}} \;\; s^{\mathrm{NE}} \;\; \Delta x s_x^{\mathrm{SW}} \;\; \Delta x s_x^{\mathrm{SE}} \;\; \Delta x s_x^{\mathrm{NW}} \;\; \Delta x s_x^{\mathrm{NE}} \;\; \Delta y s_y^{\mathrm{SW}} \;\; \Delta y s_y^{\mathrm{SE}} \;\; \Delta y s_y^{\mathrm{NW}} \;\; \Delta y s_y^{\mathrm{NE}} \;\; \ldots$$
$$\Delta x\Delta y s_{xy}^{\mathrm{SW}} \;\; \Delta x\Delta y s_{xy}^{\mathrm{SE}} \;\; \Delta x\Delta y s_{xy}^{\mathrm{NW}} \;\; \Delta x\Delta y s_{xy}^{\mathrm{NE}} \,\big]^{\mathrm{T}}.$$

Finally, the solution for the coefficients a_{ij} is $\alpha = \boldsymbol{A}^{-1}x$, with

$$\boldsymbol{A}^{-1} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 \\
-3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 \\
9 & -9 & -9 & 9 & 6 & 3 & -6 & -3 & 6 & -6 & 3 & -3 & 4 & 2 & 2 & 1 \\
-6 & 6 & 6 & -6 & -3 & -3 & 3 & 3 & -4 & 4 & -2 & 2 & -2 & -2 & -1 & -1 \\
2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
-6 & 6 & 6 & -6 & -4 & -2 & 4 & 2 & -3 & 3 & -3 & 3 & -2 & -1 & -2 & -1 \\
4 & -4 & -4 & 4 & 2 & 2 & -2 & -2 & 2 & -2 & 2 & -2 & 1 & 1 & 1 & 1
\end{bmatrix}. \tag{36}$$

Appendix B. Adaptive time-stepping

The adaptive time-stepping procedure is an *a priori* estimation of the next time-step increment Δt before solving of the velocity. The next time-step increment is determined, according to the specified maximum Courant number $C_{\max}$. The temporal variation of the velocity during the next time-step is approximated with a first-order Taylor expansion:

$$\hat{u}\big(t^{n+1}\big) = u\big(t^n\big) + \Delta t \left.\frac{\mathrm{d}u}{\mathrm{d}t}\right|_{t^n} + \underset{\Delta t\to 0}{\mathrm{O}}\big(\Delta t^2\big), \tag{37}$$

where $\hat{u}(t^{n+1})$ is the *a priori* estimate of the velocity at the next the time step.

In order to avoid overshooting of the maximum Courant number, the maximal allowable time-step increment is restricted with the two conditions:

$$\Delta t\big|u\big(t^n\big)\big| \le C_{\max}\Delta x,$$
$$\Delta t\big|\hat{u}\big(t^{n+1}\big)\big| \le C_{\max}\Delta x. \tag{38}$$

Three scenarios are possible, depending on the sign of the $u(t^n)$ and $\left.\frac{\mathrm{d}u}{\mathrm{d}t}\right|_{t^n}$:

1) When $u(t^n) \neq 0$ and $\left.\frac{\mathrm{d}u}{\mathrm{d}t}\right|_{t^n} = 0$, the maximal allowable time-step increment is simply:

$$\Delta t^{\max} = \frac{\Delta x C_{\max}}{|u(t^n)|}. \tag{39}$$

2) When $u(t^n)$ and $\frac{du}{dt}|_{t^n}$ are not zero and have the same sign, the velocity $\hat{u}(t^{n+1})$ at the next time-step is larger in absolute values than the current velocity $u(t^n)$. Combining Eqs. (37) and (38) yields a second-order polynomial equation in $\Delta t^{\max}$, which has the following solution:

$$\Delta t^{\max} = \frac{-|u(t^n)| + \sqrt{u(t^n)^2 + 4\left|\frac{du}{dt}\right|_{t^n}|\Delta x C_{\max}}}{2\left|\frac{du}{dt}\right|_{t^n}|}. \tag{40}$$

3) When $u(t^n)$ and $\frac{du}{dt}|_{t^n}$ are not zero and have opposite signs, there is a possibility for the velocity to change sign during the time-step increment. It cannot be asserted *a priori* which of the current and next velocities will be largest in absolute values, yet both restrictions of the equation must be fulfilled. It leads to two restrictions on the maximal allowable time-step increment:

$$\Delta t_1^{\max} < \frac{\Delta x C_{\max}}{|u(t^n)|} \quad \text{if } |u(t^n)| > \left|\Delta t^{\max}\frac{du}{dt}\right|_{t^n}\Big|, \tag{41}$$

$$\Delta t_2^{\max} < \frac{|u(t^n)| + \sqrt{u(t^n)^2 + 4\left|\frac{du}{dt}\right|_{t^n}|\Delta x C_{\max}}}{2\left|\frac{du}{dt}\right|_{t^n}|} \quad \text{if } |u(t^n)| < \left|\Delta t^{\max}\frac{du}{dt}\right|_{t^n}\Big| \tag{42}$$

and

$$\Delta t^{\max} = \min\left(\Delta t_1^{\max}, \Delta t_1^{\max}\right) \tag{43}$$

Note that Δx in Eqs. (39) and (41) is the length of the downwind cell, while Δx in Eqs. (40) and (42) is the length of the upwind cell.

References

[1] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[2] S. Osher, R.P. Fedkiw, Level set methods: an overview and some recent results, J. Comput. Phys. 169 (2001) 463–502.
[3] J.A. Sethian, P. Smereka, Level set methods for fluid interfaces, Annu. Rev. Fluid Mech. 35 (2003) 341–372.
[4] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annu. Rev. Fluid Mech. 31 (1999) 567–603.
[5] Y.L. Zhang, K.S. Yeo, B.C. Khoo, C. Wang, 3D jet impact and toroidal bubbles, J. Comput. Phys. 166 (2001) 336–360.
[6] C.W. Hirt, B.D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 201–225.
[7] P. Colella, Multidimensional upwind methods for hyperbolic conservation laws, J. Comput. Phys. 87 (1990) 171–200.
[8] R. LeVeque, J. Randall, High-resolution conservative algorithms for advection in incompressible flow, SIAM J. Numer. Anal. 33 (1996) 627–665.
[9] O. Ubbink, R.I. Issay, A method for capturing sharp fluid interfaces on arbitrary meshes, J. Comput. Phys. 153 (1999) 26–50.
[10] Q. Zhang, On donating regions: Lagrangian flux through a fixed curve, SIAM Rev. 55 (2013) 443–461.
[11] D.L. Youngs, Time-dependent multi-material flow with large fluid distortion, in: K.W. Morton, M.J. Baines (Eds.), Numerical Methods for Fluid Dynamics, Academic Press, New York, 1982, pp. 273–285.
[12] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.J. Rider, A high-order projection method for tracking fluid interfaces in variable density incompressible flows, J. Comput. Phys. 130 (1997) 269–282.
[13] J.E. Pilliod Jr., E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2004) 465–502.
[14] V. Dyadechko, M. Shashkov, Moment-of-fluid interface reconstruction, Los Alamos report LA-UR-07-1537, 2006.
[15] I. Ginzburg, G. Wittum, Two-phase flows on interface refined grids modeled with VOF, staggered finite volumes, and spline interpolants, J. Comput. Phys. 166 (2001) 302–335.
[16] J. López, J. Hernández, P. Gómez, F. Faura, A volume of fluid method based on multidimensional advection and spline interface reconstruction, J. Comput. Phys. 195 (2004) 718–742.
[17] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection, Int. J. Numer. Methods Fluids 41 (2003) 251–274.
[18] E. Aulisa, S. Manservisi, R. Scardovelli, S. Zaleski, A geometrical area-preserving volume-of-fluid advection method, J. Comput. Phys. 192 (2003) 355–364.
[19] E. Aulisa, S. Manservisi, R. Scardovelli, S. Zaleski, Interface reconstruction with least-squares fit and split advection in three-dimensional Cartesian geometry, J. Comput. Phys. 225 (2007) 2301–2319.
[20] G.D. Weymouth, D.K.-P. Yue, Conservative Volume-of-Fluid method for free-surface simulations on Cartesian-grids, J. Comput. Phys. 229 (2010) 2853–2865.
[21] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (1998) 112–152.
[22] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the defined donating region scheme, Int. J. Numer. Methods Fluids 35 (2001) 151–172.
[23] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the stream scheme, J. Comput. Phys. 162 (2000) 1–32.
[24] J. Mencinger, I. Zun, A PLIC–VOF method suited for adaptive moving grids, J. Comput. Phys. 230 (2011) 644–663.
[25] A. Cervone, S. Manservisi, R. Scardovelli, S. Zaleski, A geometrical predictor–corrector advection scheme and its application to the volume fraction function, J. Comput. Phys. 228 (2009) 406–419.
[26] P. Liovic, D. Lakehal, A Newton–Krylov solver for remapping-based volume-of-fluid methods, SIAM J. Sci. Comput. 31 (2008) 865–889.
[27] Q. Zhang, On a family of unsplit advection algorithms for volume-of-fluid methods, SIAM J. Numer. Anal. 51 (2013) 2822–2850.
[28] Q. Zhang, P.L.-F. Liu, A new interface tracking method: the polygonal area mapping method, J. Comput. Phys. 227 (2008) 4063–4088.
[29] Q. Zhang, A. Fogelson, Fourth-order interface tracking in two dimensions via an improved polygonal area mapping method, SIAM J. Sci. Comput. 36 (2014) A2369–A2400.
[30] Q. Zhang, Highly accurate Lagrangian flux calculation via algebraic quadratures on spline-approximated donating regions, Comput. Methods Appl. Mech. Eng. 264 (2013) 191–204.

[31] R. Comminal, J. Spangenberg, J.H. Hattel, A two-phase flow solver for incompressible viscous fluids, using a pure streamfunction formulation and the volume of fluid technique, Defect Diffus. Forum 348 (2014) 9–19.
[32] J. O'Rourke, Computational Geometry in C, 2nd ed., Cambridge University Press, New York, 1998.
[33] G. Černe, S. Petelin, I. Tiselj, Numerical errors of the volume-of-fluid interface tracking algorithm, Int. J. Numer. Methods Fluids 38 (2002) 329–350.
[34] Y. Renardy, M. Renardy, PROST: a parabolic reconstruction of surface tension for the volume-of-fluid method, J. Comput. Phys. 183 (2002) 400–421.
[35] S.V. Diwakar, S.K. Das, T. Sundararajan, A quadratic spline based interface (QUASI) reconstruction algorithm for accurate tracking of two-phase flows, J. Comput. Phys. 228 (2009) 9107–9130.
[36] T.G. Vignesh, S. Bakshi, Noniterative interface reconstruction algorithms for volume of fluid method, Int. J. Numer. Methods Fluids 73 (2013) 1–18.
[37] J. López, J. Hernández, P. Gómez, F. Faura, An improved PLIC-VOF method for tracking thin fluid structures in incompressible two-phase flows, J. Comput. Phys. 208 (2005) 51–74.
[38] E. Aulisa, S. Manservisi, R. Scardovelli, A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows, J. Comput. Phys. 188 (2003) 611–639.
[39] E. Aulisa, S. Manservisi, R. Scardovelli, A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking, J. Comput. Phys. 197 (2004) 555–584.
[40] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, S. Zaleski, Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows, J. Comput. Phys. 152 (1999) 423–456.
[41] P. Liovic, M. Rudman, J.L. Liow, D. Lakehal, D.B. Kothe, A 3D unsplit-advection volume tracking algorithm with planarity-preserving interface reconstruction, Comput. Fluids 35 (2006) 1011–1032.
[42] G.H. Miller, P. Colella, A conservative three-dimensional Eulerian method for coupled solid–fluid shock capturing, J. Comput. Phys. 183 (2002) 26–82.
[43] J. Hernández, J. López, P. Gómez, C. Zanzi, F. Faura, A new volume of fluid method in three dimensions—part I: multidimensional advection method with face-matched flux polyhedral, Int. J. Numer. Methods Fluids 58 (2008) 897–921.
[44] T. Maric, H. Marschall, D. Bothe, voFoam – a geometrical volume of fluid algorithm on arbitrary unstructured meshes with local dynamic adaptive mesh refinement using OpenFOAM, arXiv:1305.3417, 2013.
[45] J. López, J. Hernández, Analytical and geometrical tools for 3D volume of fluid methods in general grids, J. Comput. Phys. 227 (2008) 5939–5948.
[46] M. Owkes, O. Desjardins, A computational framework for conservative, three-dimensional, unsplit, geometric transport with application to the volume-of-fluid (VOF) method, J. Comput. Phys. 270 (2014) 587–612.
[47] L. Jofre, O. Lehmkuhl, J. Castro, A. Oliva, A 3-D Volume-of-Fluid advection method based on cell-vertex velocities for unstructured meshes, Comput. Fluids 94 (2014) 14–29.
[48] F. Lekien, J. Marsden, Tricubic interpolation in three dimensions, Int. J. Numer. Methods Eng. 63 (2005) 455–471.
[49] V. Le Chenadec, H. Pitsch, A 3D unsplit forward/backward volume-of-fluid approach and coupling to the level set method, J. Comput. Phys. 233 (2013) 10–33.
[50] R. Scardovelli, S. Zaleski, Analytical relations connecting linear interfaces and volume fractions in rectangular grids, J. Comput. Phys. 164 (2000) 228–237.
[51] S. Mosso, C. Garasi, R. Drake, A smoothed two- and three-dimensional interface reconstruction method, Comput. Vis. Sci. 12 (2009) 365–381.
[52] X. Yang, A.J. James, Analytic relations for reconstructing piecewise linear interfaces in triangular and tetrahedral grids, J. Comput. Phys. 214 (2006) 41–54.

D. Paper–IV

Comminal, R, Spangenberg, J & Hattel, JH 2015, 'Robust simulations of viscoelastic flows at high Weissenberg numbers with the streamfunction-log–conformation formulation', *Journal of Non-Newtonian Fluid Mechanics*, vol. 223, pp. 37–61.

Journal of Non-Newtonian Fluid Mechanics 223 (2015) 37–61

Contents lists available at ScienceDirect

Journal of Non-Newtonian Fluid Mechanics

journal homepage: http://www.elsevier.com/locate/jnnfm

Robust simulations of viscoelastic flows at high Weissenberg numbers with the streamfunction/log-conformation formulation

Raphaël Comminal [a,*], Jon Spangenberg [a,b], Jesper Henri Hattel [a]

[a] Department of Mechanical Engineering, Technical University of Denmark, Produktionstorvet, 2800 Kgs. Lyngby, Denmark
[b] Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ 08544, USA

ARTICLE INFO

Article history:
Received 18 March 2015
Received in revised form 13 May 2015
Accepted 15 May 2015
Available online 23 May 2015

Keywords:
Incompressible viscoelastic flow
Streamfunction formulation
Log-conformation representation
Lid-driven cavity
High Weissenberg number
Elastic instability

ABSTRACT

A new streamfunction/log-conformation formulation of incompressible viscoelastic flows is presented. The log-conformation representation guaranties the positive-definiteness of the conformation tensor and obviates the high Weissenberg number problem. The streamfunction is defined as a vector potential of the velocity field, and provides a pressureless formulation of the conservation laws, which automatically enforces the incompressibility. The resulting numerical method is free from velocity–pressure decoupling errors, and can achieve stable calculations for large Courant numbers, which improve the robustness and the efficiency of the solver. The two-dimensional flow of an Oldroyd-B fluid inside the lid-driven cavity is simulated for a large range of Weissenberg numbers. The numerical results demonstrate the second-order accuracy of our scheme, and our solutions are in good agreement with the available data from the literature for Weissenberg number 3 and below. Finally, the simulations at higher Weissenberg numbers 5 and 10 reveal a structural mechanism that sustains quasi-periodic elastic instabilities arising at the upstream corner of the moving lid.

1. Introduction

The simulation of viscoelastic flows has a wide range of applications in natural science and engineering. Viscoelastic fluids exhibit different flow features as compared to purely viscous fluids, such as *turbulence drag reduction* [1], *elastic instabilities* [2,3], and *elastic turbulence* at low Reynold numbers [4,5]. These viscoelastic effects are magnified in micro-fluidics [6]. To cite but a few examples of applications in polymer processing: viscoelasticity plays a role in the well-known *extrudate swelling* [7,8] and *fracture melt* [9,10] phenomena seen in extrusion. In multilayer co-extrusion, elastic forces can generate *interfacial waves* [11,12], and secondary flows known as *elastic rearrangement* [13,14], that deform the interface. Thus, the development of accurate numerical tools for the simulation of viscoelastic flows is important from both fundamental and practical points of view.

Phenomenologically, viscoelastic materials have stress responses that do not only depend on their current deformations but also on their deformation history. Thus, time is a key parameter; and viscoelastic materials have different stress responses depending on the time-scales of the deformation. They typically exhibit stress relaxation, creep deformation and shape memory. The relative effect of viscoelasticity is usually quantified by the dimensionless *Weissenberg number*[1] *Wi*, which is given by the product of a characteristic relaxation time λ of the material and a characteristic deformation rate $\dot{\varepsilon}$ of the flow:

$$Wi = \lambda\dot{\varepsilon}. \tag{1}$$

Another dimensionless quantity, the *Deborah number*, is sometimes used for flows with non-constant stretch history (i.e. deformation rate). It is defined as the ratio between the characteristic relaxation time and the time-scale of observation t_c:

$$De = \lambda/t_c. \tag{2}$$

For a critical discussion about the different uses of *Wi* and *De*, we refer to [15].

We specifically focus on the case of polymeric materials, where the viscoelastic effects arise from the *kinetics* of the polymer chains. The relaxation of diluted short-chain polymers is captured by the Rouse model, while the kinetics of entangled long-chain and branched polymers are described in the tube theories [16,17] that include the reptation mechanism. Multi-scale micro–macro simulations, where the viscoelastic stress is directly computed

* Corresponding author.
E-mail address: raphael.comminal@gmail.com (R. Comminal).

[1] Also noted *We* in some publication; not to be confused with the dimensionless Weber number which compares the fluid's inertia to its surface tension.

http://dx.doi.org/10.1016/j.jnnfm.2015.05.003
0377-0257/

from kinetics theories, using Brownian dynamics methods or stochastic models, can be found in [18]. However, the most popular approach to simulate macroscopic flows remains *continuum mechanics*. The derivation of continuum constitutive equations from the kinetics theories requires closure approximations, but in most of the numerical simulations with practical applications, the constitutive model and the material properties are rather determined from rheological measurements. The extension of constitutive models estimated by curve-fitting of rheometric flows (e.g. in simple shear or uniaxial extensional), to the description of more complex flows, should however be approached cautiously [19]. Many of the popular constitutive models can predict unphysical features [20], referred to as *constitutive instabilities*, which are classified into two types: the Hadamard instability and the dissipative instability [21,22]. The first is related to the elastic response under sudden or high frequency wave disturbances. The latter is related to unbound free energy or rate of energy dissipation with increasing deformations. Thus, appropriate constitutive models should be chosen carefully.

In addition to the physical flow instabilities and the unphysical constitutive instabilities, the numerical simulations of viscoelastic flows are also prone to *numerical instabilities*, due to the stiff hyperbolic nature of the differential constitutive equations. The numerical difficulty in solving the constitutive equations, referred to in the literature as the *high Weissenberg number problem* (HWNP), has been one of the main challenges of computational rheology for more than two decades [23–25]. The HWNP manifests itself by a loss of reliability of the simulations, leading to a blowup of the numerical values. It is triggered when the Weissenberg number (or the Deborah number) reaches a critical value, which depends on the flow problem, the spatial discretization, and the numerical algorithm. Common stabilization strategies have consisted in enhancing the elliptic operator related to the viscous stresses, as for instance with the both-side-diffusion scheme [26]. However, this stabilization approach degrades the accuracy of transient solutions [26], and values of magnitude $\sim$1 are still considered as high values of the Weissenberg numbers. Kupferman [27] has subsequently showed that the smoothness of the numerical solution can be destroyed by divergence-free stress perturbations, which do not couple back to the velocity field. Therefore, it was rather suggested to add stress diffusion terms into the constitutive models, in order to change the equation type from hyperbolic to parabolic [28], and ultimately to improve the numerical stability. Moreover, a small amount of isotropic stress diffusion has a physical justification, because of the Brownian motion of the polymer molecules. Although excessive artificial stress diffusion negatively affects the accuracy of numerical simulations [29,30], small values do not alter the flow dynamics while enhancing numerical stability by maintaining a smooth and bounded stress solution [31,32].

Lately, the origin of the HWNP was identified. First, it is acknowledged [33–35] that the loss of positive-definiteness of the *conformation tensor*—an internal variable representing the macromolecular configuration of the polymer chains, which should be symmetric positive-definite, by definition—is a precursor of the HWNP. Secondly, Fattal and Kupferman [36,37] have shown that numerical instabilities arise from the under-resolution of spatial stress profiles, which is problematic, since the solutions of viscoelastic flows tend to have stress boundary layers with large variations in the stress gradients and exponential stress profiles near geometrical singularities [38]. Failure to represent the stress gradients accurately (for instance due to polynomial interpolations of exponential profiles) results in underestimations of the convective stress fluxes, which are compensated by a multiplication of the stress growth rate, eventually leading to a numerical blowup.

Based on these observations, several techniques have been developed in order to enforce the positive-definiteness of the conformation tensor by construction. Today, the most popular method is the log-conformation representation of Fattal and Kupferman [35,36], which consists in a change of variable using the matrix-logarithm of the conformation tensor. In addition to preserving the positive-definiteness, the log-conformation representation improves the representation of large stress gradients, since it linearizes the exponential stress profiles. The log-conformation representation has been implemented in various finite-volume [39–42] and finite-element [37,43–48] codes. Alternatively, Vaithianathan and Collins [49] have presented two other schemes that guaranty the positive-definiteness of the conformation tensor, based on matrix decompositions. The first scheme uses the eigendecomposition, and evolves the eigenvalues and eigenvectors of the conformation tensor; the second scheme uses the Cholesky decomposition. Lozinski and Owens [50] and Balci et al. [51] have both proposed the square root conformation representation, based on the matrix decomposition of the conformation tensor with its principal square root (i.e. its unique positive-definite square root). Other transformations preserving the positive-definiteness of the conformation have been discussed in [52]. However, among all these different techniques, the log-conformation representation is the only one that linearizes the exponential stress profile, while for instance the square root conformation representation only reduces the exponential growth [53]. Thus, the log-conformation representation has superiority in representing stress profiles with exponential variations.

The novelty of our work lies in the combination of the log-conformation representation alongside with the pure streamfunction formulation of the Navier–Stokes equations, which was introduced in [54] as an improvement of the vorticity-based methods. The pure streamfunction formulation has demonstrated its robustness and efficiency in the case of Newtonian fluid mechanics [55–57]. Our paper aims at presenting additional arguments showing the particular benefits of the pure streamfunction formulation in the case of viscoelastic flows. The concept of streamfunctions has already been used by Chilcott and Rallison [58], and in the stream-tube algorithm [59,60], to solve static equilibriums in viscoelastic fluids. In contrast, the herein pure streamfunction method solves the entire flow dynamics. The pure streamfunction formulation is a pressureless method which automatically enforces the fluid incompressibility. Thus, the numerical errors are significantly reduced, as compared to projection methods which use velocity and pressure as primary variables. In essence, the pure streamfunction formulation has a more implicit nature, which improves the numerical stability. Thus, the combination of the log-conformation representation and the pure streamfunction formulation yields a robust numerical method for the simulation of viscoelastic flow. The present work focuses on two-dimensional viscoelastic flows; however, the pure streamfunction formulation can also be implemented in three-dimensions, see for instance [61].

The rest of the paper is organized as follows: Section 2 presents the governing equations of the viscoelastic flows and recalls the log-conformation representation of the constitutive models. Section 3 gives a critical discussion about the standard decoupling methods used for the velocity–pressure flow formulation. The pure streamfunction formulation of the conservation laws is presented in Section 4. The numerical methods used in our two-dimensional implementation of the streamfunction/log-conformation scheme are described in Section 5. The numerical results in Section 6 illustrate the second-order accuracy of our scheme, and demonstrate the stability of the method in solving the viscoelastic lid-driven cavity flow, at high Weissenberg numbers. In spite of the simple geometry, the fluid inside the lid-driven cavity is subjected to complex flow features, combining shear and extensional deformations. Therefore it is a good benchmark test to

access the robustness of the transient viscoelastic flow solver. Finally, the main results are summarized in the conclusion.

2. Governing equations

2.1. Standard differential model

In the standard Eulerian flow formulation, the dynamics of incompressible flows is governed by the conservation of mass and linear momentum:

$$\nabla \cdot \boldsymbol{u} = 0, \tag{3}$$

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}\right) = -\nabla p + \nabla \cdot \boldsymbol{\tau}, \tag{4}$$

where $\boldsymbol{u}$ is the velocity vector, ρ is the density of the fluid, p is an isostatic pressure field, and $\boldsymbol{\tau}$ is the stress response of the material to the deformation, determined from constitutive relations. The material response is decomposed with the *solvent–polymer stress splitting* technique, into a purely viscous contribution $\boldsymbol{\tau}_S$, which corresponds to the instantaneous response of the solvent, and a viscoelastic contribution $\boldsymbol{\tau}_P$, accounting for the memory effects of the polymers:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_S + \boldsymbol{\tau}_P, \tag{5}$$

with

$$\boldsymbol{\tau}_S = 2\eta_S \boldsymbol{D}, \tag{6}$$

$$\boldsymbol{\tau}_P = G_0 f_S(\boldsymbol{c}), \tag{7}$$

where $\eta_S = \beta\eta_0$ is the solvent viscosity, $G_0 = (1-\beta)\eta_0/\lambda$ is the elastic moduli,

$$\boldsymbol{D} = \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right)/2 \tag{8}$$

is the rate of deformation tensor, and $f_S(\boldsymbol{c})$ is a strain function expressed in terms of the conformation tensor $\boldsymbol{c}$. The steady-state viscosity η_0, the relaxation time λ, and the retardation ratio $\beta \in [0,1]$ are the material parameters of linear viscoelasticity.

The conformation tensor represents the spatial configuration, or the orientation, of the polymer chains. It is defined as the statistical average of the dyadic product of the end-to-end connector vector $\vec{\boldsymbol{Q}}$ of the polymer chain carrying the elastic load:

$$\boldsymbol{c} \equiv \left\langle \vec{\boldsymbol{Q}}\vec{\boldsymbol{Q}}^T \right\rangle = \int \vec{\boldsymbol{Q}}\vec{\boldsymbol{Q}}^T \psi\left(\vec{\boldsymbol{Q}}, t\right) d\vec{\boldsymbol{Q}}, \tag{9}$$

where $\psi\left(\vec{\boldsymbol{Q}}, t\right)$ is a probability distribution function following the Fokker–Planck equation. Consequently, the conformation tensor is—by definition—*symmetric positive definite*. The conformation tensor is normalized such that, at equilibrium, it is equal to the identity matrix $\boldsymbol{I}$. The conformation tensor is governed by an additional partial-differential equation that describes the relaxation of the internal deformation, with the following generic form:

$$\overset{\nabla}{\boldsymbol{c}} = -\frac{1}{\lambda} f_R(\boldsymbol{c}), \tag{10}$$

where $f_R(\boldsymbol{c})$ is the relaxation function, and $\overset{\nabla}{\boldsymbol{c}}$ is the upper-convected time derivative, defined as:

$$\overset{\nabla}{\boldsymbol{c}} \equiv \frac{\partial \boldsymbol{c}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{c} - \left(\nabla \boldsymbol{u} \cdot \boldsymbol{c} + \boldsymbol{c} \cdot \nabla \boldsymbol{u}^T\right). \tag{11}$$

The contravariant terms (those which are inside the parenthesis) in Eq. (11) are necessary in order for the model to fulfill the principle of *material frame-invariance* (also referred to as "material objectivity"). The functions $f_S(\boldsymbol{c})$ and $f_R(\boldsymbol{c})$ are polynomials (generally first and second order) of the conformation tensor, where their coefficients g_k can depend on the first and second invariants of $\boldsymbol{c}$:

$$f(\boldsymbol{c}) = \sum_k g_k \boldsymbol{c}^k. \tag{12}$$

Within this generic formulation, Eqs. (7), (10)–(12) can encompass a large choice of differential constitutive models derived from molecular theories [62]. However, Eq. (10) excludes the models using the Gordon–Schowalter convected time derivative, since it can produce unphysical behavior [22,63–65]. Table 1 provides the strain and relaxation functions corresponding to the most popular differential constitutive models, where L^2, b, ε and α are material parameters of the nonlinear viscoelasticity.

In the present work, we solely address the simulation of Oldroyd-B fluids, which have the following strain and relaxation functions:

$$f_S(\boldsymbol{c}) = f_R(\boldsymbol{c}) = \boldsymbol{c} - \boldsymbol{I}, \tag{13}$$

where $\boldsymbol{I}$ is the identity matrix. The Oldroyd-B model corresponds to the limiting case of linear-viscoelasticity, for most of the non-linear models. It is known for making unrealistic predictions in pure extensional deformations, due to the dissipative constitutive instability. Nevertheless, it is a relevant model for benchmarking the numerical solver, because it is more prone to numerical instabilities than the—more realistic—non-linear viscoelastic models. In practice, the non-linearity in the constitutive models enhances the stability of the flows, and improves the numerical convergence of the solvers; see for instance the numerical investigations in [34].

2.2. The log-conformation representation

In our algorithm, the HWNP is remedied with the log-conformation representation of Fattal and Kupferman [35,36]. The generic viscoelastic model (10)–(12) is more general than the constitutive equations originally considered in [35], but it is equally applicable to the log-conformation representation, which consists in a change of variable in the term of the matrix-logarithm of the conformation tensor

$$\boldsymbol{\Psi} = \log \boldsymbol{c}. \tag{14}$$

Fattal and Kupferman have proven that the velocity gradient $\nabla \boldsymbol{u}$ can be decomposed as

Table 1
Expressions of the recoverable strain functions and the relaxation functions for popular differential constitutive models.

Constitutive models	$f_S(\boldsymbol{c})$	$f_R(\boldsymbol{c})$
Oldroyd-B	$\boldsymbol{c} - \boldsymbol{I}$	$\boldsymbol{c} - \boldsymbol{I}$
FENE-P	$\frac{\boldsymbol{c}}{1 - tr(\boldsymbol{c})/L^2} - \boldsymbol{I}$	$\frac{\boldsymbol{c}}{1 - tr(\boldsymbol{c})/L^2} - \boldsymbol{I}$
FENE-CR	$\frac{\boldsymbol{c} - \boldsymbol{I}}{1 - tr(\boldsymbol{c})/L^2}$	$\frac{\boldsymbol{c} - \boldsymbol{I}}{1 - tr(\boldsymbol{c})/L^2}$
Exponential PTT	$\boldsymbol{c} - \boldsymbol{I}$	$\exp\left(\varepsilon\, tr(\boldsymbol{c} - \boldsymbol{I})\right)(\boldsymbol{c} - \boldsymbol{I})$
Linear PTT	$\boldsymbol{c} - \boldsymbol{I}$	$(1 + \varepsilon\, tr(\boldsymbol{c} - \boldsymbol{I}))(\boldsymbol{c} - \boldsymbol{I})$
Giesekus	$\boldsymbol{c} - \boldsymbol{I}$	$\alpha\boldsymbol{c}^2 + (1 - 2\alpha)\boldsymbol{c} - (1 - \alpha)\boldsymbol{I}$

$$\nabla \boldsymbol{u} = \boldsymbol{\Omega} + \boldsymbol{E} + \boldsymbol{N}\boldsymbol{c}^{-1}, \tag{15}$$

where $\boldsymbol{\Omega}$ and $\boldsymbol{N}$ are pure rotation (anti-symmetric) matrices, and $\boldsymbol{E}$ is a pure extension (symmetric traceless) matrix. When inserting the relations (14) and (15) into Eq. (11), the term $\boldsymbol{N}\boldsymbol{c}^{-1}$ vanishes, and, the constitutive Eq. (10) finally yields the following evolution equation for the log-conformation tensor (see [35]):

$$\frac{\partial \boldsymbol{\Psi}}{\partial t} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{\Psi} - (\boldsymbol{\Omega}\boldsymbol{\Psi} - \boldsymbol{\Psi}\boldsymbol{\Omega}) - 2\boldsymbol{E} = -\frac{1}{\lambda}\exp(-\boldsymbol{\Psi}) f_R[\exp(\boldsymbol{\Psi})]. \tag{16}$$

In practice, the matrices $\boldsymbol{E}$ and $\boldsymbol{\Omega}$ are obtained from the projection of the velocity gradient into the principal base of the stress tensor. Let the eigendecomposition of the conformation tensor in a two-dimensional flow be

$$\boldsymbol{c} = \boldsymbol{R}\begin{pmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}\boldsymbol{R}^T, \tag{17}$$

where λ_1 and λ_2 are the eigenvalues and $\boldsymbol{R}$ the orthogonal matrix containing the eigenvectors. Then the change of base of the velocity gradient is

$$\boldsymbol{R}^T(\nabla \boldsymbol{u})\boldsymbol{R} = \begin{pmatrix}m_{11} & m_{12}\\ m_{21} & m_{22}\end{pmatrix}. \tag{18}$$

Finally, the pure extension and rotation matrices in the reference base are obtained as:

$$\boldsymbol{E} = \boldsymbol{R}\begin{pmatrix}m_{11} & 0\\ 0 & m_{22}\end{pmatrix}\boldsymbol{R}^T, \tag{19}$$

and

$$\boldsymbol{\Omega} = \boldsymbol{R}\begin{pmatrix}0 & \zeta\\ -\zeta & 0\end{pmatrix}\boldsymbol{R}^T, \tag{20}$$

where

$$\zeta = \frac{m_{12}+m_{21}}{\lambda_2-\lambda_1}. \tag{21}$$

In the special case where the reference base is the principal base of the stress tensor, i.e. $\boldsymbol{c}$ is already diagonal, then $\boldsymbol{E} = \boldsymbol{D}$ and $\boldsymbol{\Omega} = \boldsymbol{0}$.

In summary, the viscoelastic stress solver finds the solution of Eq. (16) and recovers the conformation tensor with the matrix-exponential of $\boldsymbol{\Psi}$:

$$\boldsymbol{c} = \exp(\boldsymbol{\Psi}). \tag{22}$$

The formula (22) enforces by construction the symmetric positive-definiteness of the conformation tensor. Note that the matrix-logarithm and the matrix-exponential are tensor operations, where the logarithm and the exponent are applied to the eigenvalues of a tensor, for instance:

$$\log(\boldsymbol{c}) \equiv \boldsymbol{R}\begin{pmatrix}\log\lambda_1 & 0\\ 0 & \log\lambda_2\end{pmatrix}\boldsymbol{R}^T. \tag{23}$$

3. Comments on the velocity–pressure decoupling methods

In the standard flow formulation, the evolution of the velocity vector is determined from the momentum Eq. (4), while the continuity Eq. (3) is simply a kinematic constraint. Unlike the velocity vector and the conformation tensor, the isostatic pressure field does not have an evolution equation; it is determined through the coupling of Eqs. (3) and (4). The isostatic pressure is defined as the isotropic stress component of the material:

$$p \equiv -\frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma}) \tag{24}$$

where

$$\boldsymbol{\sigma} \equiv -p\boldsymbol{I} + \boldsymbol{s} \tag{25}$$

is the Cauchy stress, $\boldsymbol{I}$ is the identity matrix, and $\boldsymbol{s}$ is the deviatoric stress component. Since the entire Cauchy stress must contribute to the conservation of momentum, Eqs. (4), (24) and (25) imply that

$$\boldsymbol{s} = \boldsymbol{\tau}. \tag{26}$$

For inelastic (purely viscous) materials, $\boldsymbol{\tau}_P = 0$ and $\boldsymbol{\tau} = \boldsymbol{\tau}_S$ is by definition traceless, hence the consistency between Eqs. (6), (24)–(26). However, in the case of viscoelastic materials, the polymer stress component $\boldsymbol{\tau}_P$ is in general not traceless, i.e. it also contributes to the isotropic stress. Thus, inserting relations (5)–(7), (25) into Eq. (26) yields

$$\boldsymbol{\sigma} = -p\boldsymbol{I} + 2\eta_S\boldsymbol{D} + G_0 f_S(\boldsymbol{c}), \tag{27}$$

which is inconsistent with the pressure definition (24), since $tr[f_S(\boldsymbol{c})] \neq 0$ in general. Fortunately, the pressure field in an incompressible flow is not a thermodynamic state variable (as opposed to the case of compressible flows); therefore this inconsistency does not affect the dynamics of the flow [66]. Here, the pressure variable rather acts mathematically as a Lagrange multiplier that ensures that the continuity constraint (3) is locally fulfilled [67]. This could explain why this inconsistency has received relatively little attention in the computation rheology literature, yet.

Oliveira [68] has proposed a traceless viscoelastic stress formulation for the Oldroyd-B model, where an augmented pressure $\tilde{p}$ is defined as the true isotropic stress component

$$\tilde{p} \equiv p - \frac{1}{3}\mathrm{tr}(\boldsymbol{\tau}_P), \tag{28}$$

and a new constitutive equation is derived for the deviatoric part of the viscoelastic stress

$$\tilde{\boldsymbol{\tau}}_P \equiv \boldsymbol{\tau}_P - \frac{1}{3}\mathrm{tr}(\boldsymbol{\tau}_P)\boldsymbol{I}. \tag{29}$$

Although both formulations are equivalent from a dynamical point of view, the results of Oliveira shows that the traceless viscoelastic stress formulation improves the convergence of the numerical algorithm, i.e. it is numerically more stable. Oliveira tested the traceless viscoelastic stress formulation in two geometries, with and without a geometrical singularity, for Weissenberg numbers ranging from 0.3 to 1.5. In these numerical experiments, the traceless viscoelastic stress formulation saved between 24% and 84% of the total number of iterations, as compared to the standard pressure formulation. Moreover, the higher the Weissenberg number, the larger the gain in computational efficiency. This result suggests that the polymer contribution to the Cauchy stress degrades the iterative convergence of the velocity–pressure–extra-stress segregated solver algorithms. In general, an iterative segregated algorithm that is used to solve more than two coupled variables can be prone to convergence difficulties, if two of the variables have competing effects on the third. This is the case of the standard velocity–pressure–extra-stress decoupled approach, where both the pressure p and the isotropic part of the elastic stress $tr(\boldsymbol{\tau}_P)/3$ interact with the velocity field as Lagrange multipliers of the continuity constraint. It can be seen in the simulations of Kwon [69] that fluctuations in the viscoelastic stress cause fluctuations in the pressure field. Therefore, it is reasonable to assume that numerical errors propagate to the different variables because of the segregated algorithm. Moreover, the absence of direct coupling in the equations between the pressure and viscoelastic stress, which could have stabilized the algorithm, negatively impacts the convergence of the decoupled velocity–pressure–extra-stress methods.

Xue et al. [26] have investigated the efficiency of various decoupled velocity–pressure algorithms of the SIMPLE (Semi-Implicit

Method for Pressure Linked Equations) family, in the context of viscoelastic flows. In their numerical experiments, Xue et al. have considered the 1D start-up and decay of plane Poiseuille flow with an Oldroyd-B fluid, at $Re = 0.05$. They showed that above a threshold value of the Courant number, the transient solution of the viscoelastic flow is prone to unphysical oscillations, although all the governing equations are temporarily discretized with unconditionally stable schemes. These numerical instabilities come from the semi-implicit velocity–pressure decoupling. Xue et al. have determined that the SIMPLEST and PISO schemes, which are the most accurate algorithms of the SIMPLE family, are also the most stable (i.e. they have higher threshold Courant number), in both the Newtonian and the viscoelastic cases. The threshold Courant number for stable calculations also depends on the temporal discretization scheme. For instance, the threshold values for the SIMPLEST algorithm were $C = 2$ for the Crank–Nicolson scheme, and $C = 3.3$ for the two-level backward differentiation formula. However, the test case considered in Xue et al. is relatively simple: the 1D nature of the problem disables the coupling between some of the velocity and stress components, which are zeros in this case. Moreover, the constant and uniform solution for the pressure gradient along the channel, which is entirely determined by the pressure difference imposed at the boundaries, produces a particularly mild coupling between the velocity and the pressure. Therefore, lower threshold Courant numbers are to be expected in less favorable cases of complex flows.

The decoupled algorithms of the SIMPLE family can be seen as special cases of the general projection (or fractional-step) methods [70]. The velocity and pressure unknowns are linearly coupled in the discretized momentum and continuity equations. Thus, the coupled flow problem can be formulated as a matrix system containing the linear system of equations:

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{S} \\ \mathbf{0} \end{bmatrix}, \tag{30}$$

where the matrix operator $\mathbf{A}$ is the Jacobi matrix of the velocity for the momentum equations, $\mathbf{G}$ is the gradient operator of the pressure inside the momentum equation, $\mathbf{D}$ is the divergence operator of the velocity inside the continuity equation, $\mathbf{S}$ is the source term vector of the momentum equation (containing the boundary conditions, the polymer stress contributions, external forces and additional explicit terms), and $\mathbf{0}$ is a null vector or null matrix. The Jacobi operator $\mathbf{A}$ has the following form:

$$\mathbf{A} = \frac{1}{\Delta t}\left[\mathbf{I} + \Delta t\left(\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right)\right], \tag{31}$$

where $\mathbf{I}$ is an identity matrix, $\mathbf{N}$ is the linearized operator of the convective terms, $\mathbf{L}$ is the operator for the viscous stress (similar to a Laplace operator), Δt is the time-step size, and Re is the Reynolds number of the flow. Direct solving of the matrix system (30) is a difficult task because of the zero block diagonal matrix, reflecting the absence of pressure terms in the continuity equation, which produces an ill-conditioned matrix system. The standard projection method resolves the velocity–pressure coupling with an *LU-decomposition* of the matrix system:

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \tag{32}$$

Therefore, intermediate variables $\mathbf{u}^*$ and $\mathbf{p}^*$ can be introduced, such that:

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{u}^* \\ \mathbf{p}^* \end{bmatrix} = \begin{bmatrix} \mathbf{S} \\ \mathbf{0} \end{bmatrix}, \tag{33}$$

$$\begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{u}^* \\ \mathbf{p}^* \end{bmatrix}, \tag{34}$$

which leads to the following sequential (of fractional-step) decoupled solution:

$$\begin{cases} \mathbf{A}\mathbf{u}^* = \mathbf{S}, \\ \mathbf{D}\mathbf{A}^{-1}\mathbf{G}\mathbf{p} = \mathbf{D}\mathbf{u}^*, \\ \mathbf{u} = \mathbf{u}^* - \mathbf{A}^{-1}\mathbf{G}\mathbf{p}. \end{cases} \tag{35}$$

In this type of decoupling algorithm, solving the vector of pressure unknowns $\mathbf{p}$ is often the bottleneck of the computational cost. At this point the projection method is exact, however, the inverse matrix $\mathbf{A}^{-1}$ is unknown and expensive to compute. Fractional step methods introduce an approximation of the inverse matrix as $\mathbf{B} \approx \mathbf{A}^{-1}$. Following the analysis of Perot [67], approximate inverses of arbitrary order of accuracy can be constructed in the form of series expansion:

$$\mathbf{B} = \sum_{k=1}^{N} \Delta t^k (-1)^{k-1} \left(\mathbf{N} - \frac{\beta}{Re}\mathbf{L}\right)^{k-1} + O(\Delta t^{N+1}), \tag{36}$$

where N is the order of accuracy of the approximate inverse matrix $\mathbf{B}$. Such approximations introduce decoupling errors

$$e = \|\mathbf{A}\mathbf{B} - \mathbf{I}\| \tag{37}$$

that add up to the other sources of numerical errors (discretization, iterative solver, round-off, etc.), and which translates into non-conservation of the momentum. Decoupling errors are generally considered acceptable if the approximate inverse $\mathbf{B}$ has an order of accuracy N equal or higher than the order of accuracy of the discretization. However, a dimensional analysis of the decoupling errors shows that:

$$e \propto \Delta t \left|\frac{\beta}{Re} - 1\right|^N. \tag{38}$$

Consequently, the decoupling errors become large at low Re numbers (and especially when $Re < \beta/2$), unless Δt gets very small. This can also be explained intuitively: at low Re, the flow is driven by the pressure gradients (since the inertia effects are negligible); therefore the coupling between viscous stress and the pressure gradient is strong. Furthermore, relation (38) shows that beyond a critical case where $Re < \beta/2$, the decoupling errors become very large, and get worse with higher order N of the decoupling method. In this case, limiting the decoupling errors by low time-step sizes would become immensely onerous. Therefore, the decoupled velocity–pressure approach used in sequential solvers becomes inefficient and inaccurate at low Re number. Moreover, since the continuity equation is solved indirectly, the mass conservation is usually not fulfilled to the machine precision, but its accuracy depends on the conditioning number of the matrix product $\mathbf{BC}$.

In conclusion, the decoupled velocity–pressure methods, in the SIMPLE family or in the more general fractional step methods, present several limitations in the solving of viscous and viscoelastic flow. To summarize:

- the competing effects of the pressure and the extra-stress terms degrade the iterative convergence of segregated solvers,
- the decoupled velocity–pressure approach produces additional numerical errors that can trigger unphysical oscillations,
- the splitting technique requires restrictions of the time-step size, which become very severe at low Re,
- the discrete continuity is not fulfilled exactly, potentially producing artificial compressibility effects.

For all these reasons, the use of a pressureless flow formulation (i.e. based on the vorticity equation ensuing from the conservation of angular momentum) is justified, in order to improve the accuracy and the stability of the viscoelastic flow solvers.

4. The pure streamfunction formulation

The inherent numerical difficulties due to decoupling (or projection) methods used with the velocity–pressure flow formulation can be avoided with a pressureless formulation based on the vorticity equation. In this section, the conservation laws are reformulated in term of a streamfunction field—which is a vector potential of the velocity field—and which removes all the previously mentioned drawbacks of the decoupled velocity–pressure methods. In particular, its implicit nature improves the accuracy and stability of the calculation.

By means of the vector calculus identity $\nabla \times (\nabla p) = 0$ for any scalar field p, the pressure unknown vanishes from the conservation laws by taking the curl of the momentum Eq. (4), yielding the following equation:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \nabla \times ((\boldsymbol{u} \cdot \nabla)\boldsymbol{u}) = \nabla \times \left(\frac{\nabla \cdot \boldsymbol{\tau}}{\rho}\right), \tag{39}$$

where

$$\boldsymbol{\omega} = \nabla \times \boldsymbol{u} \tag{40}$$

is the vorticity of the flow. The curl of the inertia term (second term of the left hand side) is further treated using the following vectorial identities:

$$(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = \frac{1}{2}\nabla(\boldsymbol{u} \cdot \boldsymbol{u}) - \boldsymbol{u} \times (\nabla \times \boldsymbol{u}) \tag{41}$$

and

$$\nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) = \boldsymbol{u}(\nabla \cdot \boldsymbol{\omega}) - \boldsymbol{\omega}(\nabla \cdot \boldsymbol{u}) + (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} - (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega}, \tag{42}$$

where $\nabla \cdot \boldsymbol{\omega} = 0$ by definition, and $\nabla \cdot \boldsymbol{u} = 0$ because of the incompressibility. In addition, the curl of the purely viscous stress becomes:

$$\nabla \times \left(\frac{\nabla \cdot 2\eta_S \boldsymbol{D}}{\rho}\right) = \frac{\beta \eta_0}{\rho}\nabla^2 \boldsymbol{\omega}. \tag{43}$$

Finally combining Eqs. (39)–(43) yields the vorticity equation:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} - (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} = \frac{\eta_0}{\rho}\left[\beta \nabla^2 \boldsymbol{\omega} + \frac{1-\beta}{\lambda}\nabla \times (\nabla \cdot f_S(\boldsymbol{c}))\right]. \tag{44}$$

The vorticity equation is still coupled with the continuity Eq. (3). The incompressibility further yields a Poisson equation for the velocity:

$$\nabla^2 \boldsymbol{u} = -\nabla \times \boldsymbol{\omega}. \tag{45}$$

Various methods exist to solve velocity–vorticity flow formulations, where the vorticity equation is coupled with either the continuity Eq. (3), or the Poisson Eq. (45). However, vorticity-based methods are still subjected to decoupling errors. Moreover, the lack of natural boundary conditions for the vorticity is an additional difficulty, especially since the vorticity is created at the walls.

In the pure streamfunction formulation, the incompressibility constraint is enforced using the Helmholtz decomposition, which states that any vector field that is twice continuously differentiable in all points can be decomposed into the sum of an irrotational (curl-free) vector field and a solenoidal (divergence-free) vector field. Hence, there exits scalar and vector potentials, φ and $\boldsymbol{\Phi}$, respectively associated to the irrotational and the solenoidal velocity components, such that:

$$\boldsymbol{u} = \nabla\varphi + \nabla \times \boldsymbol{\Phi}. \tag{46}$$

These potentials are not unique, but defined up to an arbitrary constant. For an incompressible flow, the velocity field is by definition solenoidal; hence its irrotational component $\nabla\varphi$ must be zero, yielding the following equation:

$$\boldsymbol{u} = \nabla \times \boldsymbol{\Phi}. \tag{47}$$

Combining the relations (45) and (47) also yields the following constraint:

$$\boldsymbol{\omega} = -\nabla^2 \boldsymbol{\Phi} \tag{48}$$

Finally, a unique dynamical equation can be derived by replacing all the velocity and vorticity unknowns in Eq. (44) by the streamfunction unknown, using Eqs. (47) and (48).

In the two-dimensional case, the velocity vector has two in-plane components, while the vorticity and the streamfunction fields only have their out-of-plane component which is non-zeros:

$$\boldsymbol{u} = (u, v, 0)^T, \quad \boldsymbol{\omega} = (0, 0, \omega)^T, \quad \boldsymbol{\Phi} = (0, 0, \phi)^T. \tag{49}$$

Thus, the two-dimensional flow is governed by a single equation accounting for both the momentum conservation and the incompressibility constraint:

$$\frac{\partial \nabla^2 \phi}{\partial t} + \frac{\partial \phi}{\partial y}\frac{\partial \nabla^2 \phi}{\partial x} - \frac{\partial \phi}{\partial x}\frac{\partial \nabla^2 \phi}{\partial y} = \frac{\eta_0}{\rho}\left[\beta \nabla^4 \phi - \frac{1-\beta}{\lambda}\nabla \times (\nabla \cdot f_S(\boldsymbol{c}))\right], \tag{50}$$

where the term $\nabla^4\phi$ is the biharmonic operator of the streamfunction. Then, the two components of the velocity vector are recovered as:

$$\begin{aligned} u &= \partial\phi/\partial y, \\ v &= -\partial\phi/\partial x, \end{aligned} \tag{51}$$

which automatically fulfills the continuity constraint to the machine precision. Hence, the pure streamfunction flow formulation yields a fully implicit scheme, without splitting or fractional steps.

The solving of the streamfunction Eq. (50) requires boundary conditions for the streamfunction, as well as its derivatives, which are obtained from the boundary conditions of the velocity. Pressure and traction boundary conditions are not implementable as such, although tangential pressure gradients to the boundary, representing for instance a hydrostatic pressure, can be included as source terms in the streamfunction equation. Consequently, the boundary conditions for the flows which are driven by pressure differences must be specified in terms of the corresponding inflow velocity profiles and fully developed outflows. The implementation of the boundary conditions is discussed in the next section.

Finally, the pressure field can be post-processed by solving the Poisson equation, obtained by taking the divergence of the momentum equations. By virtue of the continuity, the divergence of the velocity time-derivatives and the purely viscous stresses vanishes; hence yielding:

$$\nabla^2 p = \nabla \cdot (\nabla \cdot \boldsymbol{\tau}_P) - \rho \nabla \cdot (\boldsymbol{u} \cdot \nabla \boldsymbol{u}). \tag{52}$$

Neumann pressure boundary conditions are obtained from the projection of the moment equation along the normal vector of the boundaries. In opposition to the velocity–pressure methods, here the pressure field is only calculated at the specific discrete times of interest; and it does not affect the computation of the flow dynamics.

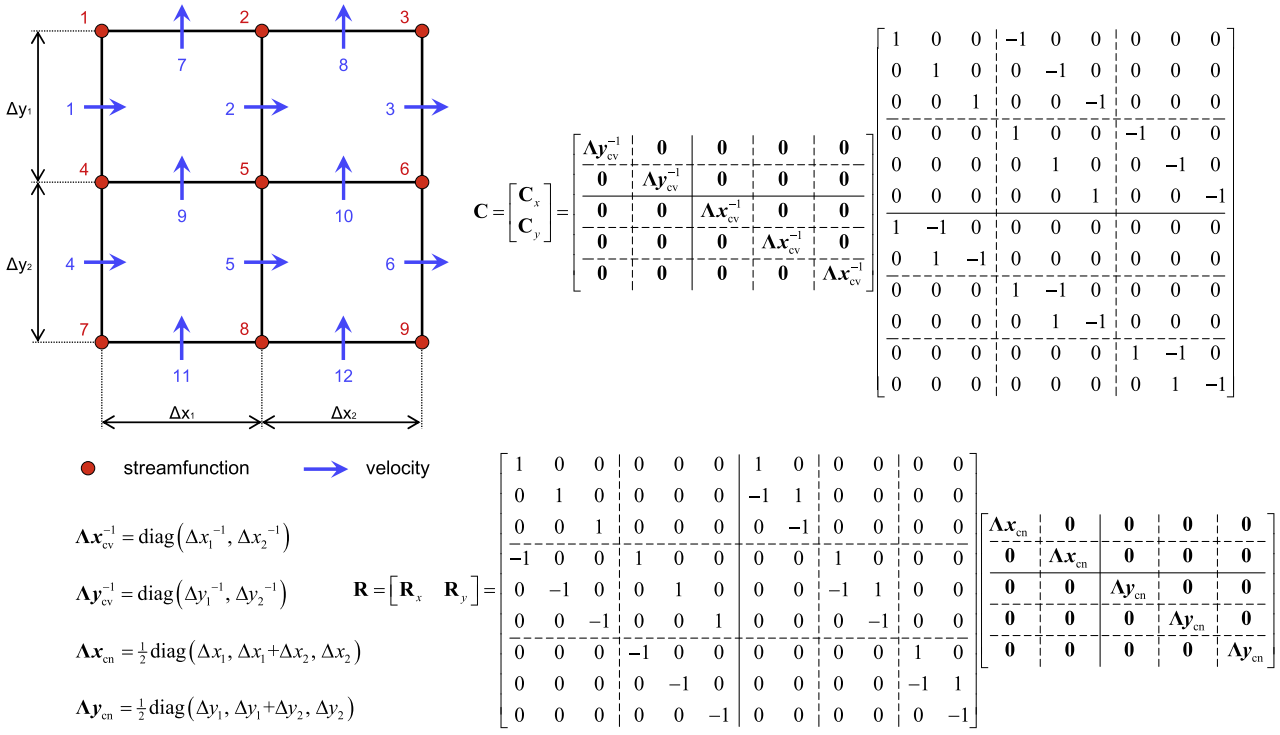


Fig. 1. Staggered variable arrangement, numbering of the unknowns, and finite-difference construction of the discrete curl operators **R** and **C**, in a 2D geometry.

5. Numerical methods

5.1. Discrete curl operators

Directly discretizing the streamfunction Eq. (50) is a rather difficult task because of the third and fourth spatial derivatives, which require high-order schemes and appropriate boundary conditions. The procedure presented in [71] alleviates this difficulty by applying discrete curl operators to the discretized momentum equations. Let the linear matrix system for the discretized momentum equations be:

$$\mathbf{Au} = \mathbf{b}, \tag{53}$$

where **u** is the solution vector of the discrete velocity variables, **A** is the Jacobi matrix of the discrete velocity variables in the discretized momentum equations, and **b** is the source term vector. Note that the pressure gradients do not need to be included in the system of Eq. (53) since they vanish with the curl transformation. A linear system of equations for the vorticity equation is first obtained by applying the discrete curl operator **R**, which relates the solution vector of the discrete velocities to the solution vector **ω** of discrete vorticity variables, to the system of discretized momentum equations:

$$\mathbf{RAu} = \mathbf{Rb}, \tag{54}$$

with

$$\boldsymbol{\omega} = \mathbf{Ru}. \tag{55}$$

The curl operator **R** results in the integration of the momentum equations along closed lines around the discrete streamfunction variables. Secondly, the linear matrix system for the vorticity equation is expressed in terms of discrete streamfunction variables, by the following change of variable:

$$\mathbf{u} = \mathbf{C}\Phi, \tag{56}$$

where **C** is the discretized curl operator relating to the discrete velocity variables to the vector of the discrete streamfunction variables **Φ**. Note that **R** and **C** are two different curl operators. Finally these transformations yield a linear matrix system expressing the vorticity equation in terms of discrete streamfunction variables:

$$\mathbf{RAC}\Phi = \mathbf{Rb}, \tag{57}$$

where the matrix products **RAC** and **Rb** are the Jacobi matrix and the source term vector, respectively.

The discrete curl operators **R** and **C** are discretized using finite-differences, taking advantage of a staggered variable arrangement on a Cartesian grid, see Fig. 1. Discrete streamfunction values are located at the grid intersections. The normal velocity components laying on the cell's edges, evaluated from Eq. (56), correspond to the face-average values. Fig. 1 shows how the discrete curl operators **R** and **C** are constructed for a Cartesian grid, but the procedure is essentially the same for unstructured grid, see [71] for more details. It is important to notice that the discrete curl operators are discretized in a way such that the null vector calculus identities $\nabla \times (\nabla a) = 0, \forall a \in \mathbb{R}$ and $\nabla \cdot (\nabla \times \mathbf{A}) = 0, \quad \forall \mathbf{A} \in \mathbb{R}^3$ are fulfilled on the discrete level:

$$\mathbf{RG} = \mathbf{0}, \tag{58}$$

$$\mathbf{DC} = \mathbf{0}, \tag{59}$$

where **G** and **D** are the discrete gradient and divergence operators in the coupled matrix system (30). These relations ensure the elimination of the pressure unknowns in the conservation equations, and the automatic enforcement of the continuity constraint on the discrete level to the machine precision.[2] Note that the line-integrations around the discrete streamfunction variables laying on the boundary

[2] In practice, there is no need to discretize the gradient and divergence operators to solve the streamfunction matrix system (57); therefore the discrete null vector calculus identities (58) and (59) are actually mental constructions.

are incomplete; hence these streamfunctions are rather determined by boundary conditions.

Essential streamfunction boundary conditions are obtained from the integration of the normal velocities over the boundary. Using the relation (47) and the Kelvin–Stokes theorem of vector calculus yields a contour integral:

$$\int\int_{\partial\Omega} \boldsymbol{u}\cdot\hat{\boldsymbol{n}}\mathrm{d}S = \int\int_{\partial\Omega} (\nabla\times\boldsymbol{\Phi})\cdot\hat{\boldsymbol{n}}\mathrm{d}S = \oint_{\Sigma} \boldsymbol{\Phi}\cdot d\boldsymbol{r} \tag{60}$$

where Σ is the contour of boundary surface $\partial\Omega$. In the two-dimensional case, the surface integral of the normal velocity becomes a line integral along the boundary, and the contour integral of the streamfunction turns into an oriented difference. Hence, the streamfunction boundary condition along the boundary segment between two points p_1 and p_2 is given by:

$$\begin{aligned}\phi_{p_1}-\phi_{p_2} &= \int_{p_1}^{p_2} v(x)\mathrm{d}x, \quad \text{along an horizontal boundary},\\ \phi_{p_2}-\phi_{p_1} &= \int_{p_2}^{p_1} u(y)\mathrm{d}y, \quad \text{along a vertical boundary}.\end{aligned} \tag{61}$$

This yields constant values of the streamfunction along the walls and the symmetry lines. At the inlet boundaries, the line integral of the normal velocity is calculated either analytically or numerically. At the outlets, where the normal velocity is an unknown, the streamfunction boundary condition is derived from the fully-developed Neumann boundary condition $\partial u_n/\partial x_n|_{\partial\Omega}=0$, which gives:

$$\partial\phi/\partial x_n|_{\partial\Omega}=0. \tag{62}$$

In addition, since the streamfunction is a potential field, a reference value needs to be specified at a reference point on the boundary, in order to set the streamfunction level:

$$\phi(\boldsymbol{x}_{\mathrm{ref}})=\phi_{\mathrm{ref}}, \quad \text{where } \boldsymbol{x}_{\mathrm{ref}}\in\partial\Omega, \tag{63}$$

for instance, with a null value on a symmetry line or a wall boundary. Finally, the boundary conditions (61)–(63) replace the incomplete equations for the boundary streamfunction variables in the matrix system (57).

The discretization of the Jacobi operator **A** of the momentum equation is treated in the next section. Natural boundary conditions for the tangential velocity to the boundaries are imposed through the source terms in the matrix system (53), with the same procedure as the one described in [67,71].

5.2. Finite-volume discretization

In this subsection, we describe the finite-volume discretization of a generic convection-diffusion equation with the form:

$$\frac{\partial\chi}{\partial t}+u_k\frac{\partial\chi}{\partial x_k}=\Gamma_\chi\frac{\partial^2\chi}{\partial x_k\partial x_k}+S_\chi, \tag{64}$$

which can either be the momentum Eq. (4) or the log-conformation Eq. (16), depending on the interpretation of χ, Γ_χ and S_χ; see Table 2. Eq. (64) is integrated over a control volume V and the divergence theorem is applied to the convective and diffusive terms:

$$\int_V \frac{\partial\chi}{\partial t}\mathrm{d}V+\oint_{\partial V}\left(u\chi-\Gamma_\chi\frac{\partial\chi}{\partial\hat{\boldsymbol{n}}}\right)\cdot\hat{\boldsymbol{n}}\mathrm{d}S=\int_V S_\chi\mathrm{d}V, \tag{65}$$

yielding a flux balance between the faces f of the control volume:

$$\frac{\partial\chi_V}{\partial t}+\frac{1}{V}\sum_{\mathrm{f}}\left(u_{\mathrm{f}}\chi_{\mathrm{f}}-\Gamma_\chi\frac{\partial\chi}{\partial x_n}\Big|_{\mathrm{f}}\right)=S_{\chi,V}, \tag{66}$$

where χ_V and $S_{\chi,V}$ are the volume-average of the transported quantity and the source term, inside the control volume. The evaluation of the fluxes requires an interpolation of the quantities χ_{f} at the faces of the control volumes, as well as their gradients $(\partial\chi/\partial x_n)|_{\mathrm{f}}$ in the normal directions. The latter gradients, which are related to the diffusive fluxes, are evaluated with the second-order central-differences of the volume-average quantities in the neighboring cells.

The face-averaged values χ_{f} are obtained with the CUBISTA scheme introduced by Alves et al. [72], which consists of a quadratic upwind scheme, with total-variation-diminishing flux limiters promoting the numerical stability. The CUBISTA scheme was specially developed to enhance the iterative convergence of viscoelastic flow solvers. For convenience, the interpolation scheme is expressed in terms of the normalized variable and space formulation, where

$$\widehat{\chi}_{\mathrm{f}}=\frac{\chi_{\mathrm{f}}-\chi_{\mathrm{UU}}}{\chi_{\mathrm{D}}-\chi_{\mathrm{UU}}}, \quad \widehat{\chi}_{\mathrm{U}}=\frac{\chi_{\mathrm{U}}-\chi_{\mathrm{UU}}}{\chi_{\mathrm{D}}-\chi_{\mathrm{UU}}} \tag{67}$$

are the normalized averaged-values of the transported quantities at the CV's face, and inside the upwind cell, respectively, and

$$\widehat{\xi}_{\mathrm{f}}=\frac{\xi_{\mathrm{f}}-\xi_{\mathrm{UU}}}{\xi_{\mathrm{D}}-\xi_{\mathrm{UU}}}, \quad \widehat{\xi}_{\mathrm{U}}=\frac{\xi_{\mathrm{U}}-\xi_{\mathrm{UU}}}{\xi_{\mathrm{D}}-\xi_{\mathrm{UU}}} \tag{68}$$

are the normalized space variables, with subscripts "D", "U" and "UU" referring to volume-average values in the downstream, first upstream and second upstream control volumes, respectively, see Fig. 2. The interpolation $\widehat{\chi}_{\mathrm{f}}\left(\widehat{\chi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{f}}\right)$ with the CUBISTA scheme reads:

$$\widehat{\chi}_{\mathrm{f}}\left(\widehat{\chi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{f}}\right)=\begin{cases}\left[1+\frac{\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}}{3\left(1-\widehat{\xi}_{\mathrm{U}}\right)}\right]\frac{\widehat{\xi}_{\mathrm{f}}}{\widehat{\xi}_{\mathrm{U}}}\widehat{\chi}_{\mathrm{U}}, & \text{if } 0<\widehat{\chi}_{\mathrm{U}}<\frac{3}{4}\widehat{\xi}_{\mathrm{U}}\\ \frac{\widehat{\xi}_{\mathrm{f}}\left(1-\widehat{\xi}_{\mathrm{f}}\right)}{\widehat{\xi}_{\mathrm{U}}\left(1-\widehat{\xi}_{\mathrm{U}}\right)}\widehat{\chi}_{\mathrm{U}}+\frac{\widehat{\xi}_{\mathrm{f}}\left(\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}\right)}{1-\widehat{\xi}_{\mathrm{U}}}, & \text{if } \frac{3}{4}\widehat{\xi}_{\mathrm{U}}\leqslant\widehat{\chi}_{\mathrm{U}}\leqslant\frac{1+2\left(\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}\right)}{2\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}}\widehat{\xi}_{\mathrm{U}}\\ 1-\frac{1-\widehat{\xi}_{\mathrm{f}}}{2\left(1-\widehat{\xi}_{\mathrm{U}}\right)}\left(1-\widehat{\chi}_{\mathrm{U}}\right), & \text{if } \frac{1+2\left(\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}\right)}{2\widehat{\xi}_{\mathrm{f}}-\widehat{\xi}_{\mathrm{U}}}\widehat{\xi}_{\mathrm{U}}<\widehat{\chi}_{\mathrm{U}}<1\\ \widehat{\chi}_{\mathrm{U}}, & \text{otherwise}\end{cases} \tag{69}$$

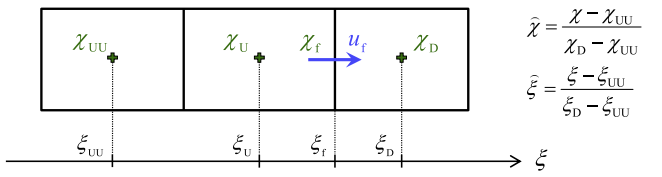


Fig. 2. Definition of the local variables and local coordinates in the upwind and downwind cells, related to interpolation $\widehat{\chi}_{\mathrm{f}}\left(\widehat{\chi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{U}},\widehat{\xi}_{\mathrm{f}}\right)$ at a faces of the control volumes.

Table 2
Interpretation of the terms χ, Γ_χ and S_χ in the generic convection-diffusion Eq. (64), for the momentum and log-conformation equations.

Equations	χ	Γ_χ	S_χ
i^{th}-component of the momentum	ρu_i	$\beta\eta_0/\rho$	$\frac{(1-\beta)\eta_0}{\lambda}[f_S(\boldsymbol{c})]_{ij}$
ij^{th}-component of the log-conformation	Ψ_{ij}	0	$(\boldsymbol{\Omega\Psi}-\boldsymbol{\Psi\Omega})_{ij}+2E_{ij}-\frac{1}{\lambda}\{\exp(-\boldsymbol{\Psi})f_R[\exp(\boldsymbol{\Psi})]\}_{ij}$

The time derivative $\partial\chi_V/\partial t$ is discretized implicitly with the two-level backward differentiation formula (BDF2):

$$\left.\frac{\partial\chi_V}{\partial t}\right|^n = \frac{1}{\Delta t^n}\left(\frac{2\varpi+1}{\varpi+1}\chi_V^n - (\varpi+1)\chi_V^{n-1} + \frac{\varpi^2}{\varpi+1}\chi_V^{n-2}\right) + O(\Delta t^2), \tag{70}$$

where the subscript n refers the value at the current time t^n, the superscripts $n-1$ and $n-2$ refer to the values at the two previous discrete times t^{n-1} and t^{n-2}, $\Delta t^n = t^n - t^{n-1}$ is the time-step increment, and $\varpi = \Delta t^n/\Delta t^{n-1}$ is the time-step increment ratio. The BDF2 scheme is *A-stable* and second-order accurate, which makes it suitable to solve stiff hyperbolic equations, with a good compromise between numerical stability and accuracy.

The quadratic non-linear terms for the inertia, in the momentum equation, are linearized using a second-order Taylor expansion:

$$\begin{aligned}(u_iu_j)^n &= (u_iu_j)^{n-1} + \left.\frac{\partial(u_iu_j)}{\partial u_k}\right|^{n-1}(u_k^n - u_k^{n-1}) + O(\Delta t^2)\\ &= u_i^n u_j^{n-1} + u_i^{n-1}u_j^n - u_i^{n-1}u_j^{n-1} + O(\Delta t^2).\end{aligned} \tag{71}$$

This approximation is suitable for approximating inertia terms for flows at low and moderate Reynolds numbers. For instance, Kim et al. [73] were able to solve accurately turbulent Newtonian flows at $Re = 4200$, with the same implicit linearization.

The divergence of the viscoelastic stress-tensor, in the source term of the momentum equations, is also approximated with finite-differences. Note that all the normal and shear components of the extra-stress are located at the center of the control volumes, similarly as the components of the log-conformation tensor. The classical finite-difference schemes take advantage of the staggered variable arrangement, by assuming linear variations of the viscoelastic extra-stresses inside the control volume (see for instance in [36]):

$$\frac{\partial\sigma_{xx}}{\partial x} = \frac{\sigma_{xx,\mathrm{E}} - \sigma_{xx,\mathrm{W}}}{(\Delta x_\mathrm{E} + \Delta x_\mathrm{W})/2}, \tag{72}$$

$$\frac{\partial\sigma_{xy}}{\partial y} = \frac{1}{2}\frac{\sigma_{xy,\mathrm{NW}} + \sigma_{xy,\mathrm{NE}} - \sigma_{xy,\mathrm{SW}} - \sigma_{xy,\mathrm{SE}}}{\Delta y_\mathrm{N} + 2\Delta y + \Delta y_\mathrm{S}}, \tag{73}$$

where the different subscripts refer to the different neighboring extra-stresses to the velocity's control volume, see Fig. 3. This classical scheme is hereinafter referred to as the *LIN* scheme. Here, we suggest a higher-order scheme assuming linear variations of the gradient of the viscoelastic extra-stress; hence quadratic variations of the viscoelastic extra-stress. This scheme is referred to as the *QUAD* scheme and the volume-averages of the stress-divergences are obtained with the following arithmetic averages:

$$\frac{\partial\sigma_{xx}}{\partial x} = \frac{\Delta x_\mathrm{E}\left.\frac{\partial\sigma_{xx}}{\partial x}\right|_\mathrm{W} + \Delta x_\mathrm{W}\left.\frac{\partial\sigma_{xx}}{\partial x}\right|_\mathrm{E}}{\Delta x_\mathrm{E} + \Delta x_\mathrm{W}}, \tag{74}$$

$$\frac{\partial\sigma_{xy}}{\partial y}\frac{(\Delta y + \Delta y_\mathrm{S})\left(\Delta x_\mathrm{E}\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{NW} + \Delta x_\mathrm{W}\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{NE}\right) + (\Delta y_\mathrm{N} + \Delta y)\left(\Delta x_\mathrm{E}\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{SW} + \Delta x_\mathrm{W}\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{SE}\right)}{(\Delta y_\mathrm{N} + 2\Delta y + \Delta y_\mathrm{S})(\Delta x_\mathrm{E} + \Delta x_\mathrm{W})}, \tag{75}$$

where

$$\left.\frac{\partial\sigma_{xx}}{\partial x}\right|_\mathrm{W} = \frac{1}{\Delta x_\mathrm{WW} + 2\Delta x_\mathrm{W} + \Delta x_\mathrm{E}}\left[(\Delta x_\mathrm{W} + \Delta x_\mathrm{WW})\frac{\sigma_{xx,\mathrm{E}} - \sigma_{xx,\mathrm{W}}}{(\Delta x_\mathrm{E} + \Delta x_\mathrm{W})/2} + (\Delta x_\mathrm{E} + \Delta x_\mathrm{W})\frac{\sigma_{xx,\mathrm{W}} - \sigma_{xx,\mathrm{WW}}}{(\Delta x_\mathrm{W} + \Delta x_\mathrm{WW})/2}\right], \tag{76}$$

$$\left.\frac{\partial\sigma_{xx}}{\partial x}\right|_\mathrm{E} = \frac{1}{\Delta x_\mathrm{W} + 2\Delta x_\mathrm{E} + \Delta x_\mathrm{EE}}\left[(\Delta x_\mathrm{E} + \Delta x_\mathrm{W})\frac{\sigma_{xx,\mathrm{EE}} - \sigma_{xx,\mathrm{E}}}{(\Delta x_\mathrm{EE} + \Delta x_\mathrm{E})/2} + (\Delta x_\mathrm{EE} + \Delta x_\mathrm{E})\frac{\sigma_{xx,\mathrm{E}} - \sigma_{xx,\mathrm{W}}}{(\Delta x_\mathrm{E} + \Delta x_\mathrm{W})/2}\right], \tag{77}$$

$$\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{NW} = \frac{\sigma_{xy,\mathrm{NW}} - \sigma_{xy,\mathrm{W}}}{(\Delta y_\mathrm{N} + \Delta y)/2}, \tag{78}$$

$$\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{NE} = \frac{\sigma_{xy,\mathrm{NE}} - \sigma_{xy,\mathrm{E}}}{(\Delta y_\mathrm{N} + \Delta y)/2}, \tag{79}$$

$$\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{SW} = \frac{\sigma_{xy,\mathrm{W}} - \sigma_{xy,\mathrm{SW}}}{(\Delta y + \Delta y_\mathrm{S})/2}, \tag{80}$$

$$\left.\frac{\partial\sigma_{xy}}{\partial y}\right|_\mathrm{SE} = \frac{\sigma_{xy,\mathrm{E}} - \sigma_{xy,\mathrm{WE}}}{(\Delta y + \Delta y_\mathrm{S})/2}, \tag{81}$$

are the finite-difference approximations of the viscoelastic extra-stress gradient, using the same staggered variable arrangement as in Fig. 3.

The nonlinearities in the constitutive equations, due to the material-derivative and the matrix-exponential of the log-conformation tensor, are treated explicitly. As a result, each evolution equation for a variable χ_V^n is discretized into the algebraic form

$$a_0^\chi\chi_V^n = \sum_\mathrm{nb} a_\mathrm{nb}^\chi\chi_\mathrm{nb}^n + b^\chi, \tag{82}$$

where χ_nb^n are unknown values in the neighboring control volumes, $a_0^\chi, a_\mathrm{nb}^\chi$ and b^χ are coefficients containing explicit expressions of the solutions of the previous time-step. The contributions of the source terms are included in b^χ. The explicit representation of the covariant term $\boldsymbol{\Omega\Psi} - \boldsymbol{\Psi\Omega}$ decouples the three components of the

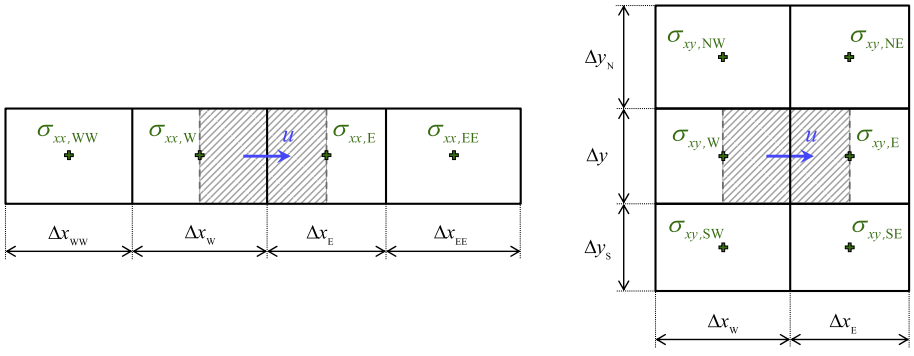


Fig. 3. Staggered variable arrangement used for the finite-difference approximation of the divergence of the viscoelastic extra-stress inside the hatched control volume.

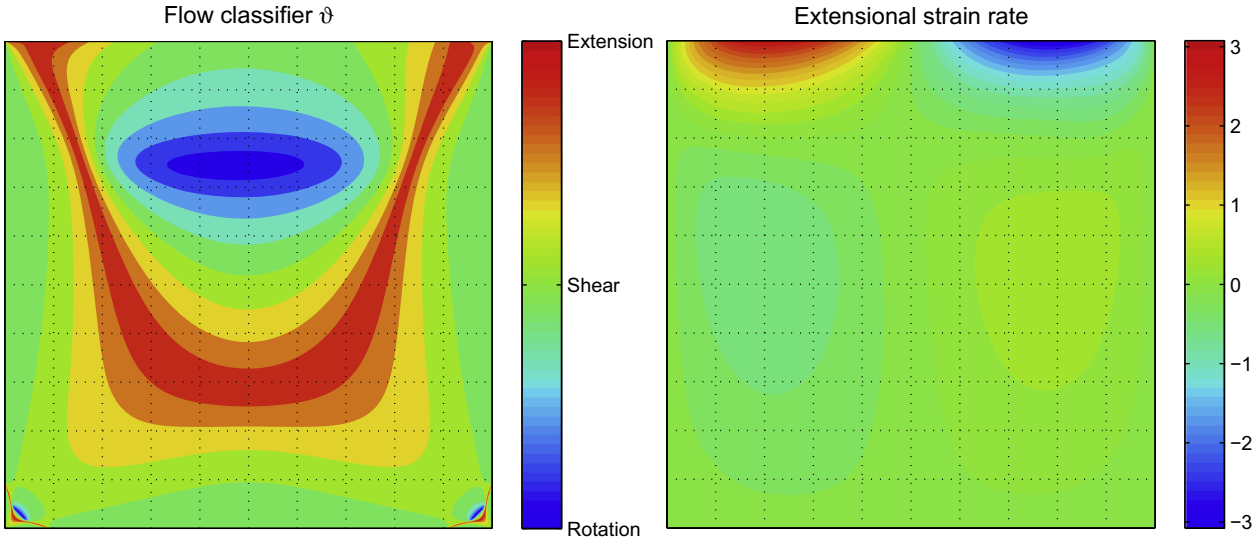


Fig. 4. Flow classifier ϑ (left) and extensional strain rate $\dot{\varepsilon}$ (right) of the steady-state solution, for $Wi = 0.25$.

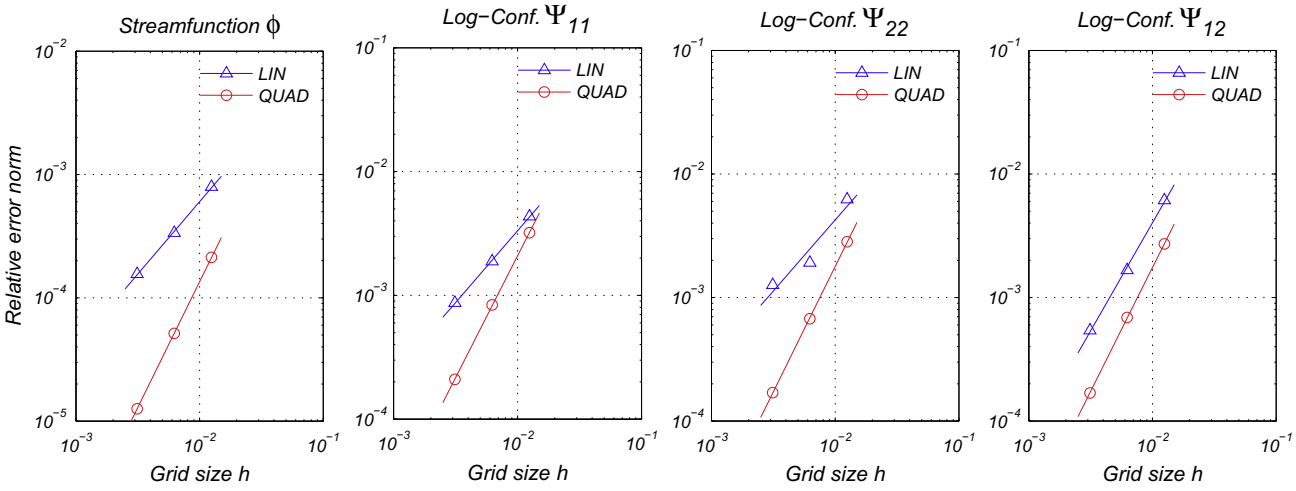


Fig. 5. Asymptotic convergence of the spatial relative error norm, for the *LIN* and *QUAD* stress interpolation schemes.

log-conformation tensor. Therefore, the two-dimensional stream function/log-conformation scheme yields four matrix systems: one for the streamfunction field, and one for each of the three components of the log-conformation tensor that are solved in parallel. Finally, the streamfunction and the log-conformation tensor are solved sequentially, with successive substitution iterations until the convergence of the nonlinearities.

5.3. Solution algorithm

At the beginning of nth time-step, provisional guessed solution vectors $\mathbf{\Phi}_0^n$ and $\mathbf{\Psi}_0^n$ are evaluated by linear extrapolation from the results of the previous time-steps. The provisional guessed solution for the velocity vector is $\mathbf{u}_0^n = \mathbf{C}\mathbf{\Phi}_0^n$. Then, the solution algorithm computes iteratively the components of the new log-conformation tensor and the new streamfunctions.

Since the governing equations are discretized with the implicit BDF2 scheme, there is no formal restriction on the time-step size. However, the successive substitution iterations can have convergence issues when the Courant number exceeds unity. The convergence of the solution algorithm is enhanced with the under-relaxation of the solution iterates. The log-conformation tensor contributes linearly to the streamfunction equations, thus the convergence of the log-conformation tensor results in the convergence of the streamfunction. In contrast, the velocity has a non-linear effect on the evolution of the log-conformation tensor. Therefore, the under-relaxation is only applied to the log-conformation iterates. We use an adaptive under-relaxation, the Aitken's delta-squared process, which improves the convergence rate of a series of solution vectors. At each k^{th} iteration, the Aitken's relaxation factor α_k is evaluated as:

$$\alpha_k = \alpha_{k-1}\left(1 + \frac{\left(\Delta\widetilde{\mathbf{\Psi}}_{k-1} - \Delta\widetilde{\mathbf{\Psi}}_k\right)^{\mathrm{T}} \cdot \Delta\widetilde{\mathbf{\Psi}}_k}{\left(\Delta\widetilde{\mathbf{\Psi}}_{k-1} - \Delta\widetilde{\mathbf{\Psi}}_k\right)^2}\right), \tag{83}$$

with

$$\Delta\widetilde{\mathbf{\Psi}}_k = \mathbf{\Psi}_{k-1}^n - \widetilde{\mathbf{\Psi}}_k^n, \tag{84}$$

Table 3
Spatial convergence rates with the *LIN* and *QUAD* stress interpolation schemes.

Variables	Spatial convergence rates	
	LIN interpolation scheme	*QUAD* interpolation scheme
ϕ	1.17	2.04
Ψ_{11}	1.16	1.97
Ψ_{22}	1.15	2.03
Ψ_{12}	1.75	2.00

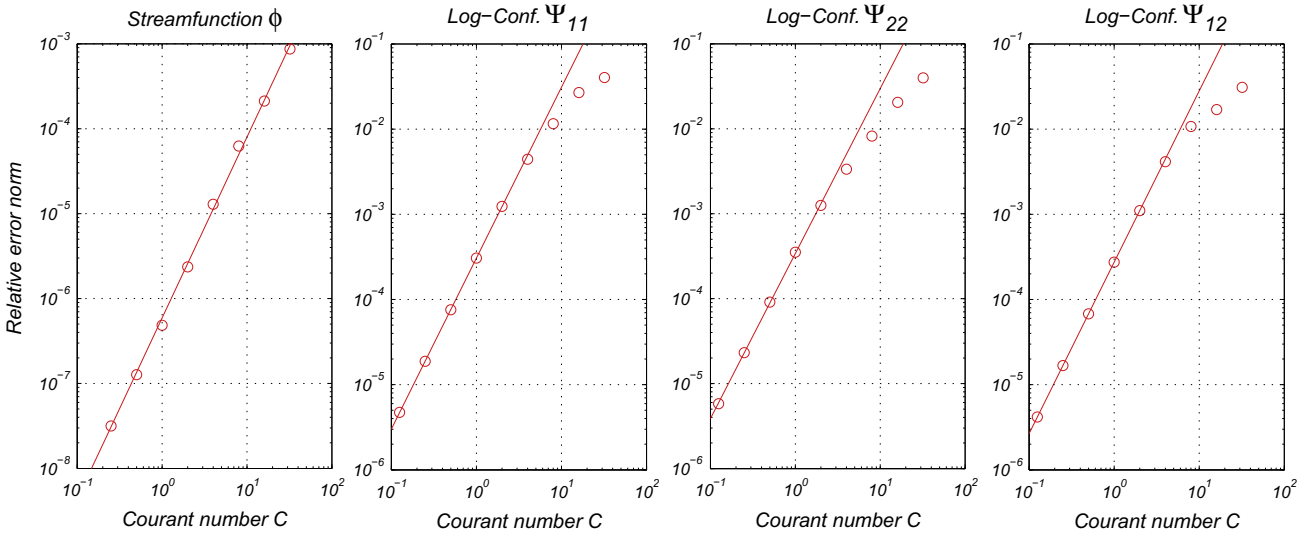


Fig. 6. Asymptotic convergence of the temporal relative error norm.

Table 4
Temporal convergence rates, for $C < 8$.

Variables	Temporal convergence rates
ϕ	2.13
Ψ_{11}	1.97
Ψ_{22}	1.96
Ψ_{12}	1.97

where $\widetilde{\boldsymbol{\Psi}}_k^n$ is the latest non-under-relaxed iterate vector of the log-conformation solution, and $\boldsymbol{\Psi}_{k-1}^n$ is the log-conformation vector of the previous under-relaxed solution iterate. In general, adaptive under-relaxation is more efficient than under-relaxation with a fixed relaxation factor. Since the Aitken's adaptive under-relaxation requires two previous solution iterates, the under-relaxation is activated from the third iteration, and the two first solution iterates are not under-relaxed.

In summary, the solution algorithm at the k^{th} iteration proceeds as follows:

1. Compute the new solution vector $\widetilde{\boldsymbol{\Psi}}_k^n$ of the discretized system of equations for each component of the log-conformation tensor:

$$\begin{aligned}&\frac{1}{\Delta t^n}\frac{2\varpi+1}{\varpi+1}\widetilde{\boldsymbol{\Psi}}_k^n+\mathbf{u}_{k-1}^n\cdot\nabla\widetilde{\boldsymbol{\Psi}}_k^n\\&=\frac{1}{\Delta t^n}\left((\varpi+1)\boldsymbol{\Psi}^{n-1}-\frac{\varpi^2}{\varpi+1}\boldsymbol{\Psi}^{n-2}\right)\\&\quad+\left(\boldsymbol{\Omega}(\mathbf{u}_{k-1}^n,\boldsymbol{\Psi}_{k-1}^n)\boldsymbol{\Psi}_{k-1}^n-\boldsymbol{\Psi}_{k-1}^n\boldsymbol{\Omega}(\mathbf{u}_{k-1}^n,\boldsymbol{\Psi}_{k-1}^n)\right)\\&\quad+2\mathbf{E}(\mathbf{u}_{k-1}^n,\boldsymbol{\Psi}_{k-1}^n)-\frac{1}{\lambda}\exp\left(-\boldsymbol{\Psi}_{k-1}^n\right)f_{\mathrm{R}}\left[\exp\left(\boldsymbol{\Psi}_{k-1}^n\right)\right],\end{aligned}\tag{85}$$

where $\mathbf{u}_{k-1}^n$ is the velocity vector of the previous solution iterate.

2. Under-relax the solution vector of the log-conformation tensor with the relaxation factor α_k evaluated from (83):

$$\boldsymbol{\Psi}_k^n=\alpha_k\widetilde{\boldsymbol{\Psi}}_k^n+(1-\alpha_k)\boldsymbol{\Psi}_{k-1}^n.\tag{86}$$

3. Recover the conformation vector $\mathbf{c}_k^n$ via the matrix-exponent of $\boldsymbol{\Psi}_k^n$:

$$\mathbf{c}_k^n=\exp\left(\boldsymbol{\Psi}_k^n\right).\tag{87}$$

4. Build-up the linear matrix system of the momentum equations discretized as:

$$\begin{aligned}&\frac{\rho}{\Delta t^n}\frac{2\varpi+1}{\varpi+1}\mathbf{u}_k^n+\rho\mathbf{N}(\mathbf{u}_k^n,\mathbf{u}^{n-1})-\eta_{\mathrm{S}}\mathbf{L}(\mathbf{u}_k^n)\\&=\frac{\rho}{\Delta t^n}\left((\varpi+1)\mathbf{u}^{n-1}-\frac{\varpi^2}{\varpi+1}\mathbf{u}^{n-2}\right)+\mathbf{D}\left(G_0f_{\mathrm{S}}(\mathbf{c}_k^n)\right).\end{aligned}\tag{88}$$

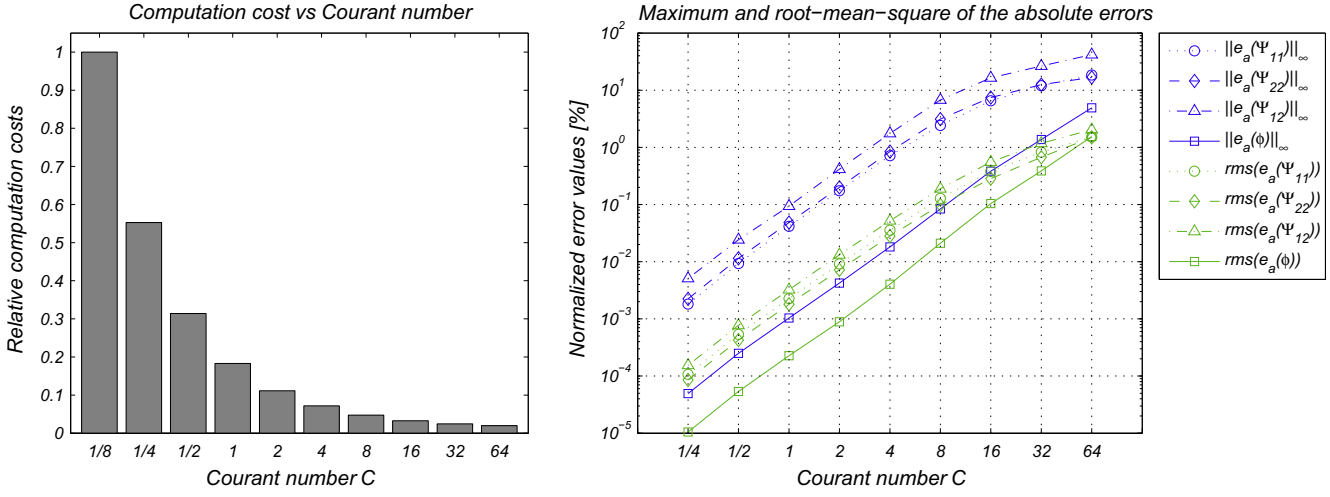


Fig. 7. Computational efficiency: the relative computation costs (left) and the normalized absolute errors (right), for different Courant numbers.

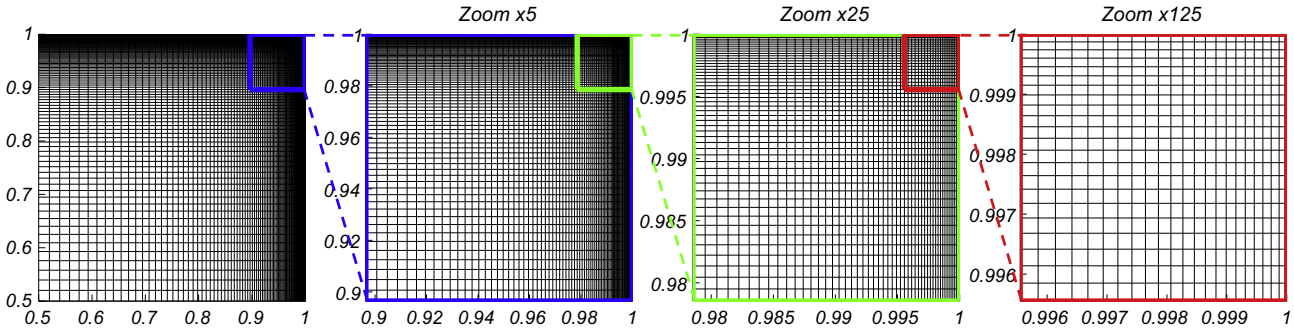


Fig. 8. Zooms on the upper right corner of the grid having a constant contraction ratio $\Delta x_{k+1}/\Delta x_k = 0.96$.

5. Apply the discrete curl transformation to the momentum matrix system and compute the new iterate of the streamfunction vector $\mathbf{\Phi}_k^n$ from the linear matrix system (57).
6. Recover the discrete velocity vector as:

$$\mathbf{u}_k^n = \mathbf{C}\mathbf{\Phi}_k^n. \tag{89}$$

7. Evaluate the scaled norm of the residual vector of the log-conformation systems of equations, using the new solution vectors $\mathbf{\Psi}_k^n$ and $\mathbf{u}_k^n$:

$$r_k = \frac{\left|\left|\mathbf{LHS}_\Psi\left(\mathbf{u}_k^n, \mathbf{\Psi}_k^n\right)\cdot\mathbf{\Psi}_k^n - \mathbf{RHS}_\Psi\left(\mathbf{u}_k^n, \mathbf{\Psi}_k^n\right)\right|\right|_2}{\left|\left|\mathbf{LHS}_\Psi\left(\mathbf{u}_k^n, \mathbf{\Psi}_k^n\right)\right|\right|_2 \cdot \left|\left|\mathbf{\Psi}_k^n\right|\right|_2 + \left|\left|\mathbf{RHS}_\Psi\left(\mathbf{u}_k^n, \mathbf{\Psi}_k^n\right)\right|\right|_2}, \tag{90}$$

where $\mathbf{LHS}_\Psi$ and $\mathbf{RHS}_\Psi$ are the new left-hand-side matrix and the right-hand-side vector for the log-conformation matrix system (85) that will be solved at the next iteration. The 2-norm of the matrix $\mathbf{LHS}_\Psi$ is approximated as its largest component in absolute value $||\mathbf{LHS}_\Psi||_2 \cong \max_{i,j}\left|(\mathbf{LHS}_\Psi)_{ij}\right|$.

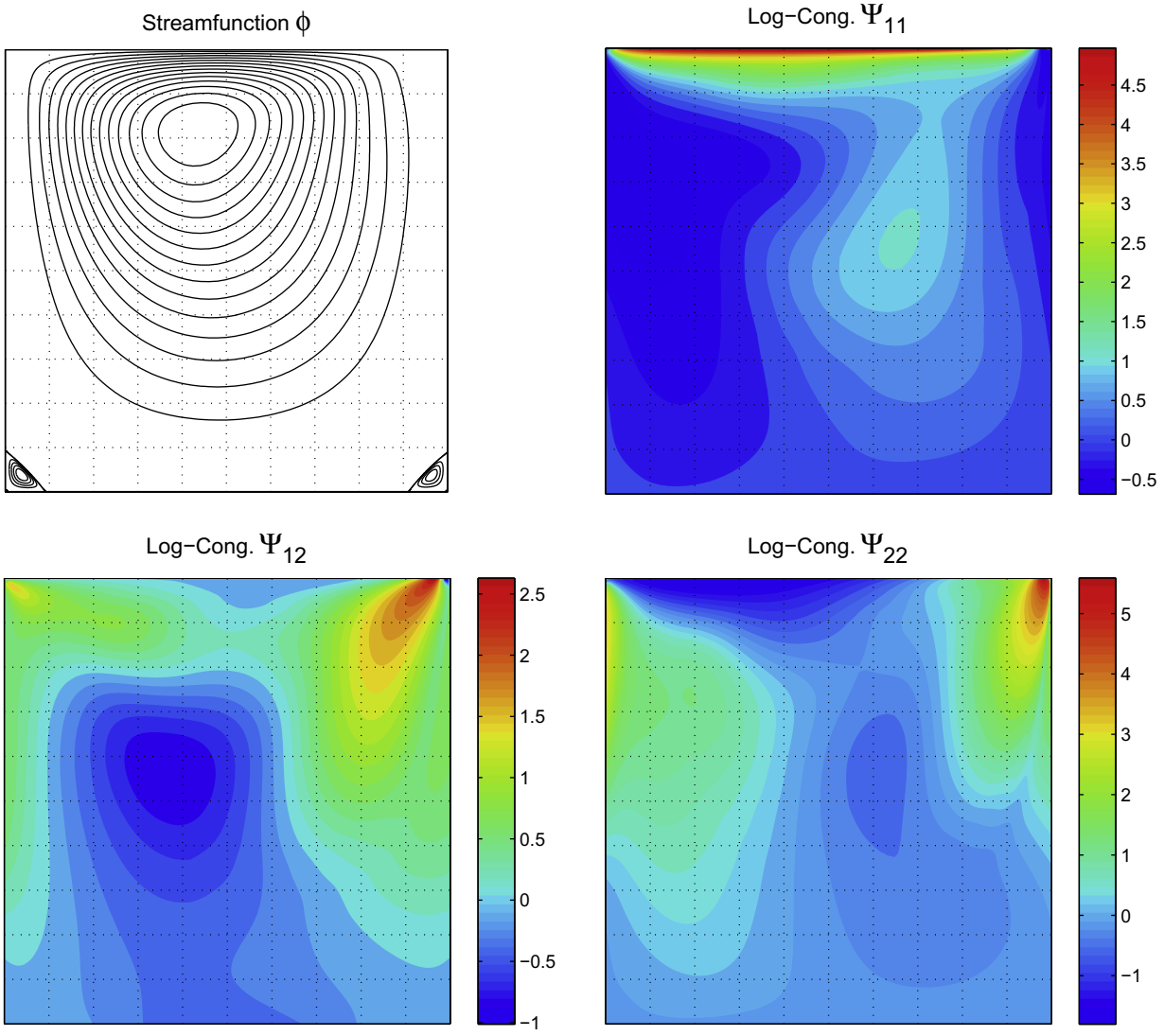


Fig. 9. The streamlines and the three components of the log-conformation tensor, at time $t = 8$, for $Wi = 1$. (The extrema of the color scales do not correspond to the extrema of the solution.) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The iteration procedure is converged when the following termination criterion is fulfilled:

$$r_k \leqslant \varepsilon_{\text{tol}}, \tag{91}$$

where ε_{tol} is a user-specified residual tolerance. The bulk of the computation resides in solving the linear matrix systems of the steps 1 and 5. The other steps of the solution algorithm consist in simple evaluations of explicit expressions.

6. Numerical results

The above scheme has been implemented to solve the regularized lid-driven cavity flow, which is a benchmark test, also solved in [35,36,45,74,75]. The geometry consists in a square cavity with a unit length $L = 1$. The flow is driven by the tangential motion of the upper wall, while the other boundaries are stationary, with the no-slip condition. The tangential velocity profile of lid is regularized as in [36]:

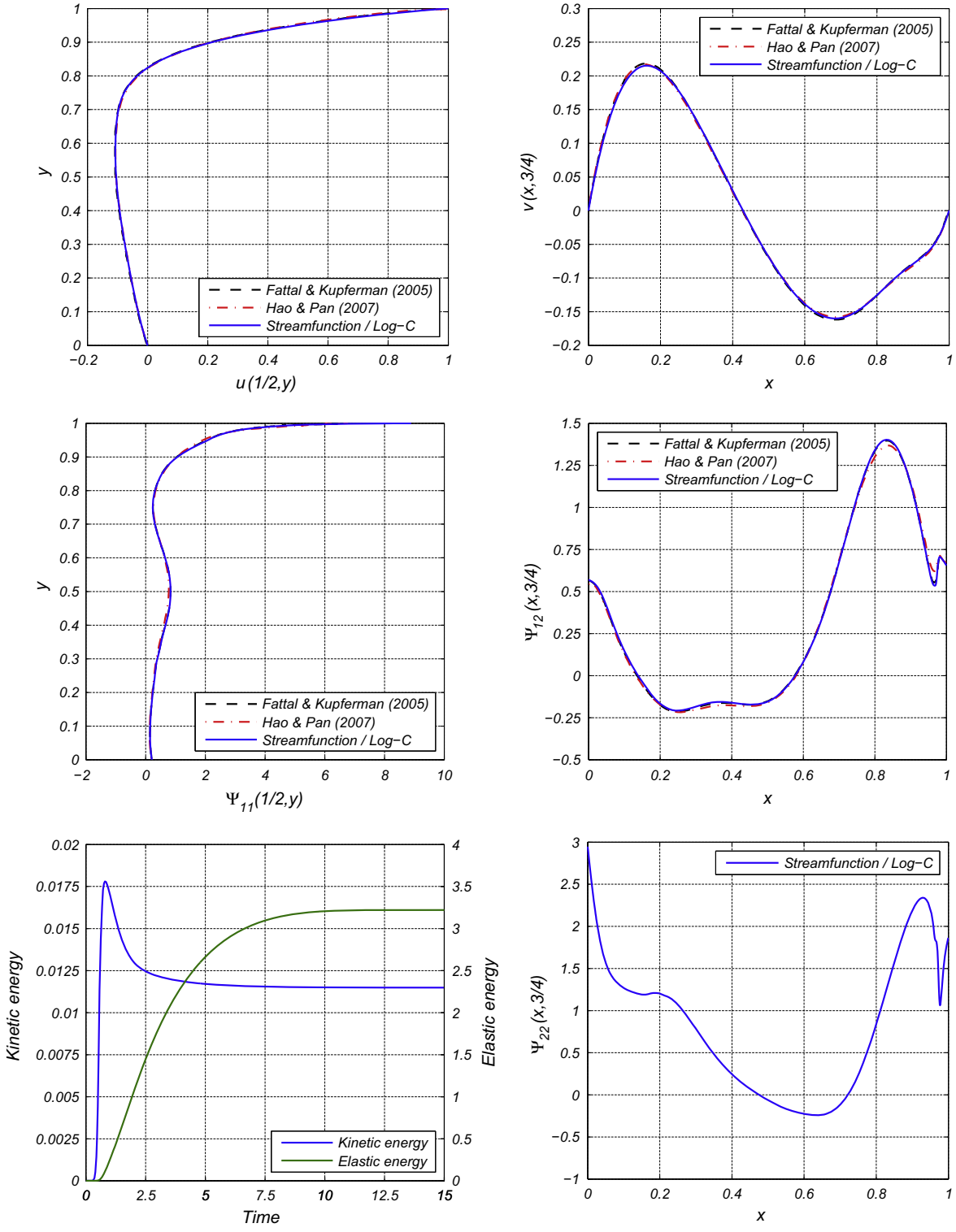


Fig. 10. Results at time $t = 8$, for $Wi = 1$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$ (compared with references [36,74]); history of the kinetic energy and elastic energy.

$$u_t(x,t) = 16x^2(1-x)^2(0.5+\tanh(8(t-0.5))/2)U, \quad x \in [0,1], \quad t>0, \tag{92}$$

in order to remove the stress singularity at the corners [76]. In this way, the stagnation points located at the corners are not subjected to any deformation rate, preventing the appearance of infinitely increasing stress gradients, which could compromise the smoothness of the solution [77]. The regularized velocity profile can be interpreted physically as the result of the fluid slippage on the moving lid, as shown by molecular dynamics simulations [78]. In addition, the hyperbolic tangent function smoothes the acceleration of the lid, and removes the sudden deformation rate at the initial time. The lid velocity has reached its nominal values when $t > 1$. The maximum lid velocity is set to $U = 1$. In this setup, the Weissenberg number is defined as $Wi = \lambda U/L \cong \lambda$.

In spite of the simple geometry, this problem is considered as particularly difficult to solve, especially at high Weissenberg numbers, since the flow exhibits complex features combining shear and extensional deformations. For $Wi = 0.25$, Fig. 4 (left) represents the flow classifier ϑ, which is defined as:

$$\vartheta \equiv \frac{1-R_S}{1+R_S}, \tag{93}$$

where

$$R_S = \frac{|\omega/2|}{\sqrt{\dot{\varepsilon}^2+\dot{\gamma}^2}} \tag{94}$$

is the persistence-of-straining parameter [79], ω is the vorticity, $\dot{\varepsilon} \equiv \partial u/\partial x$ is the extensional strain rate, and $\dot{\gamma} \equiv (\partial u/\partial y + \partial v/\partial x)/2$ is the shear rate. The flow classifier has the following properties: $\vartheta = 1 (R_S = 0)$ in pure extensional flow, $\vartheta = 0 (R_S = 1)$ in pure shear flow, and $\vartheta = -1 (R_S \to \infty)$ in pure rigid-body rotation (ϑ and R_S are indefinite when the flow is at rest or in pure translation). The magnitude of the extensional strain rate is also depicted in Fig. 4(right), for $Wi = 0.25$. Fig. 4 shows that the material successively experiences shear and extensional deformations. Moreover, the extensional rate reaches large values near the moving lid. As the cavity is a closed system (without inlets and outlets), there is a critical Weissenberg number above which the recirculating material build-up extra-stresses faster than it relaxes. Then, the flow cannot reach a steady state solution, and it exhibits elastic instabilities. In

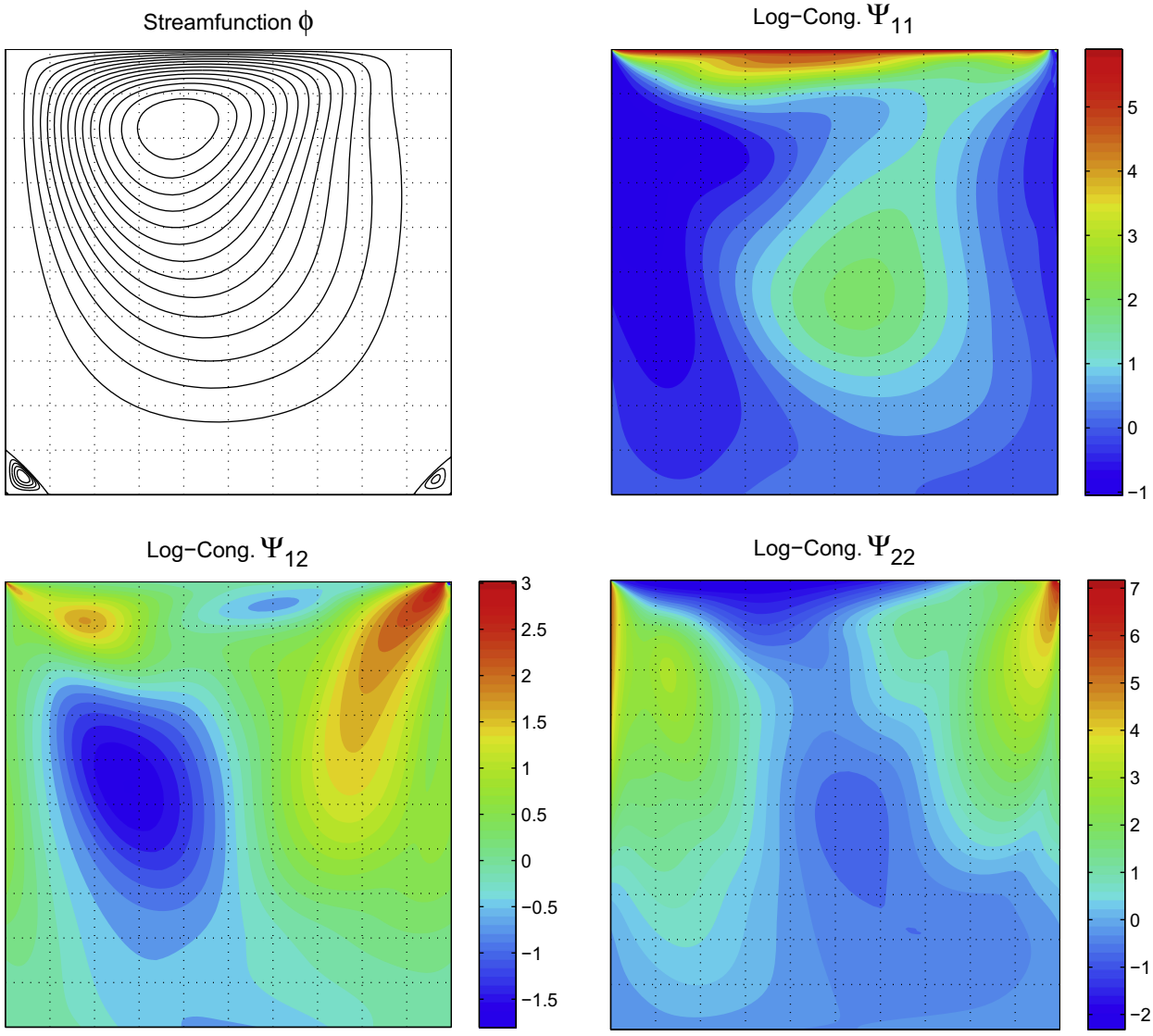


Fig. 11. The streamlines and the three components of the log-conformation tensor, at time $t = 40$, for $Wi = 2$. (The extrema of the color scales do not correspond to the extrema of the solution.) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

this scenario, the flow develops steep stress boundary layers that require a very fine resolution, otherwise the under-resolution produces numerical instabilities [36]. Moreover, in the case a Oldroyd-B fluid, which lacks finite extensibility, an unbounded stress growth occurs under steady extensional deformations, when $\dot{\varepsilon} > 1/(2\lambda)$, as a manifestation of the dissipative constitutive instability. Thus, from a numerical point of view, the simulation of the viscoelastic lid-driven cavity flow at high Weissenberg numbers is similarly challenging as its Newtonian counterpart at high Reynolds numbers.

In the following subsections, the regularized lid-driven cavity flow of the Oldroyd-B fluid is simulated for various Weissenberg numbers by varying the relaxation times. A constant retardation ratio $\beta = 1/2$ is used. The Reynold number is set to the negligible

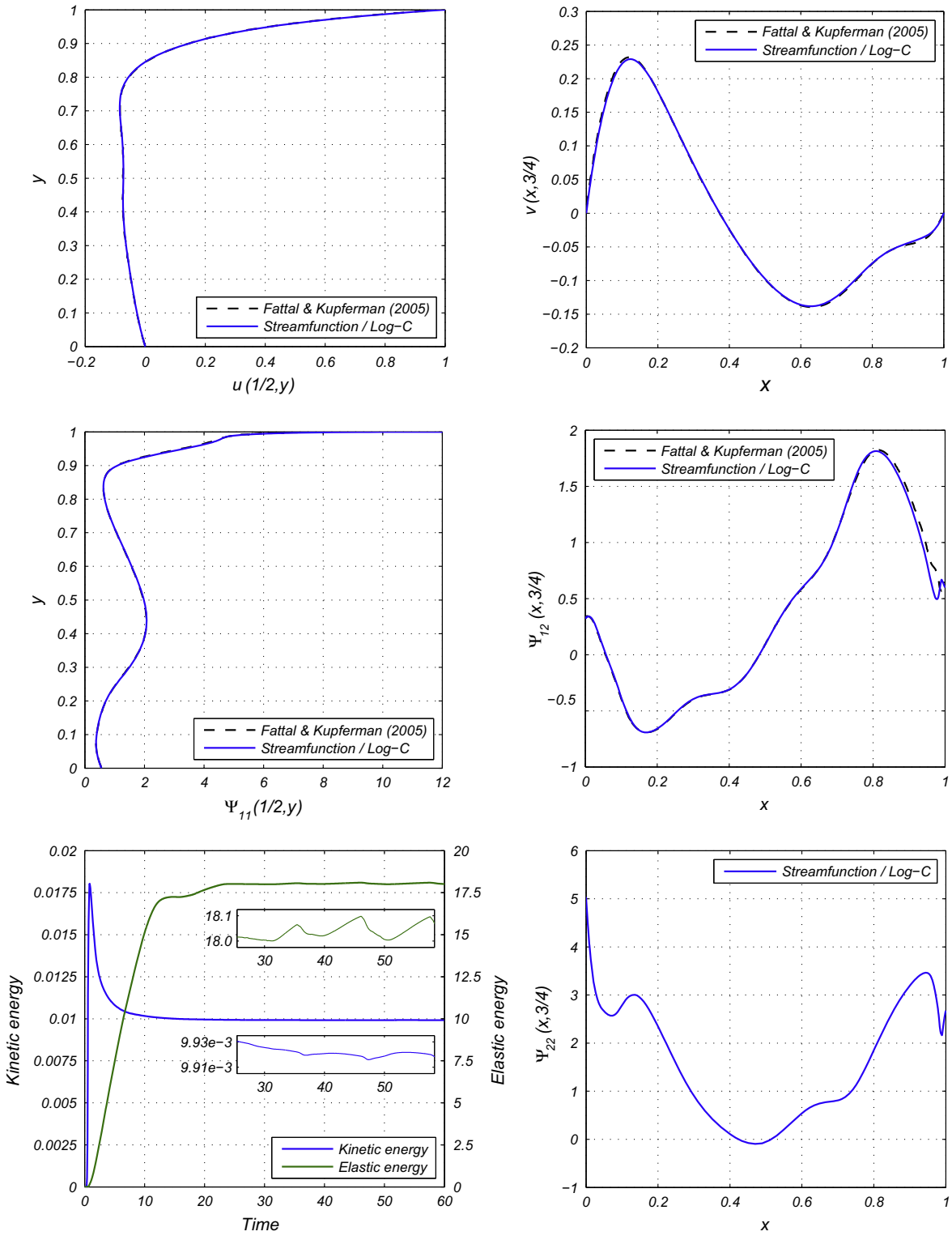


Fig. 12. Results at time $t = 40$, for $Wi = 2$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$ (compared with reference [36]); history of the kinetic energy and elastic energy.

value $Re = \rho UL/\eta_0 = 5 \cdot 10^{-4}$, such that the flow can be virtually considered as a creeping flow. To the knowledge of the authors, there is no result of viscoelastic lid-driven cavity flows reported in the literature for Weissenberg numbers above $Wi = 3$.

6.1. Order of convergence

In this subsection, where the spatial and temporal order of convergences are assessed, the residual tolerance of the iterations algorithm is set to the value $\varepsilon_{\text{tol}} = 10^{-12}$. The spatial convergence rate is estimated from the relative errors with respect to successive mesh refinements, since there is no analytical solution to evaluate the absolute errors. The simulation is carried out on Cartesian mesh with the grid spacings $h = 1/40$, $1/80$, $1/160$ and $1/320$, and with a fixed time step size $\Delta t = 10^{-2}$. The relative error is locally monitored on a sample of 81 points evenly distributed inside the cavity (whose locations correspond to the intersections of the grid lines plotted in Fig. 4). For a grid size h, the relative error $e_r(h, x_i, y_j)$ of the numerical solution $\chi_h(x_i, y_j)$ at the point with coordinates $(x_i, y_j) = (iL/10, jL/10)$ is:

$$e_r(h, x_i, y_j) = \left|\chi_{2h}(x_i, y_j) - \chi_h(x_i, y_j)\right|. \tag{95}$$

A norm for the global relative error is defined as the root-mean-square of all the local relative errors:

$$E_r(h) = \left[\frac{1}{81}\sum_{i,j=1}^{9} e_r(h, x_i, y_j)^2\right]^{1/2}. \tag{96}$$

The numerical solution has a spatial convergence rate p when its relative error norm asymptotically converges to the zero grid size limit as:

$$E_r(h) \underset{h\to 0}{\simeq} \kappa h^p + O(h^p), \tag{97}$$

where κ is a constant. The convergence rate is determined as the slope of the best fitting line of the relative errors norm plotted on logarithmic scale, see Fig. 5, where:

$$\log[E_r(h)] \simeq \log(\kappa) + p\log(h). \tag{98}$$

The spatial convergence rates for the streamfunction and the three components of the log-conformation tensor are reported in Table 3, for the *LIN* and *QUAD* stress interpolation schemes. As it can be seen,

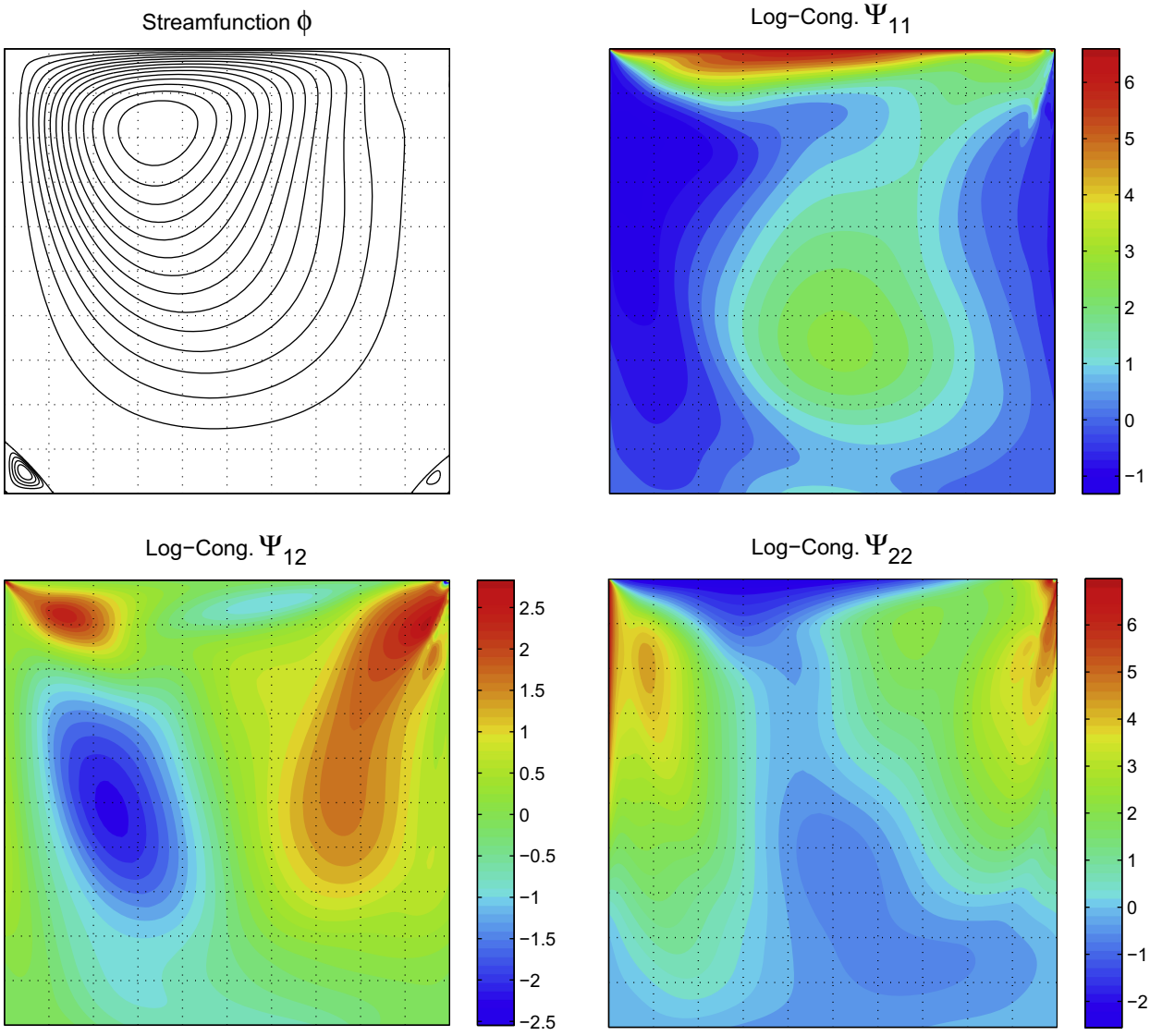


Fig. 13. The streamlines and the three components of the log-conformation tensor, at time $t = 40$, for $Wi = 3$. (The extrema of the color scales do not correspond to the extrema of the solution.) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

only the *QUAD* scheme exhibits second-order accuracy. The convergence rate of the *LIN* scheme—which is used in [36,74]—is between first and second-orders. This confirms that linear stress interpolations, in the momentum equations, degrade the second-order accuracy of the CUBISTA and the central-differencing schemes.

The temporal convergence is assessed from the transient solution at the time $t = 2$, for $Wi = 1$, such that the flow has not reached its steady-state yet. The simulation is performed for different time-step sizes, but with the same grid spacing $h = 1/160$, using the *QUAD* stress interpolation scheme. The temporal convergence rates are evaluated similarly as for the spatial convergence rates, except that the convergence is calculated with respect to refinements of the time-step size (i.e. the Courant number C), see Fig. 6. The temporal convergence is second order, for Courant

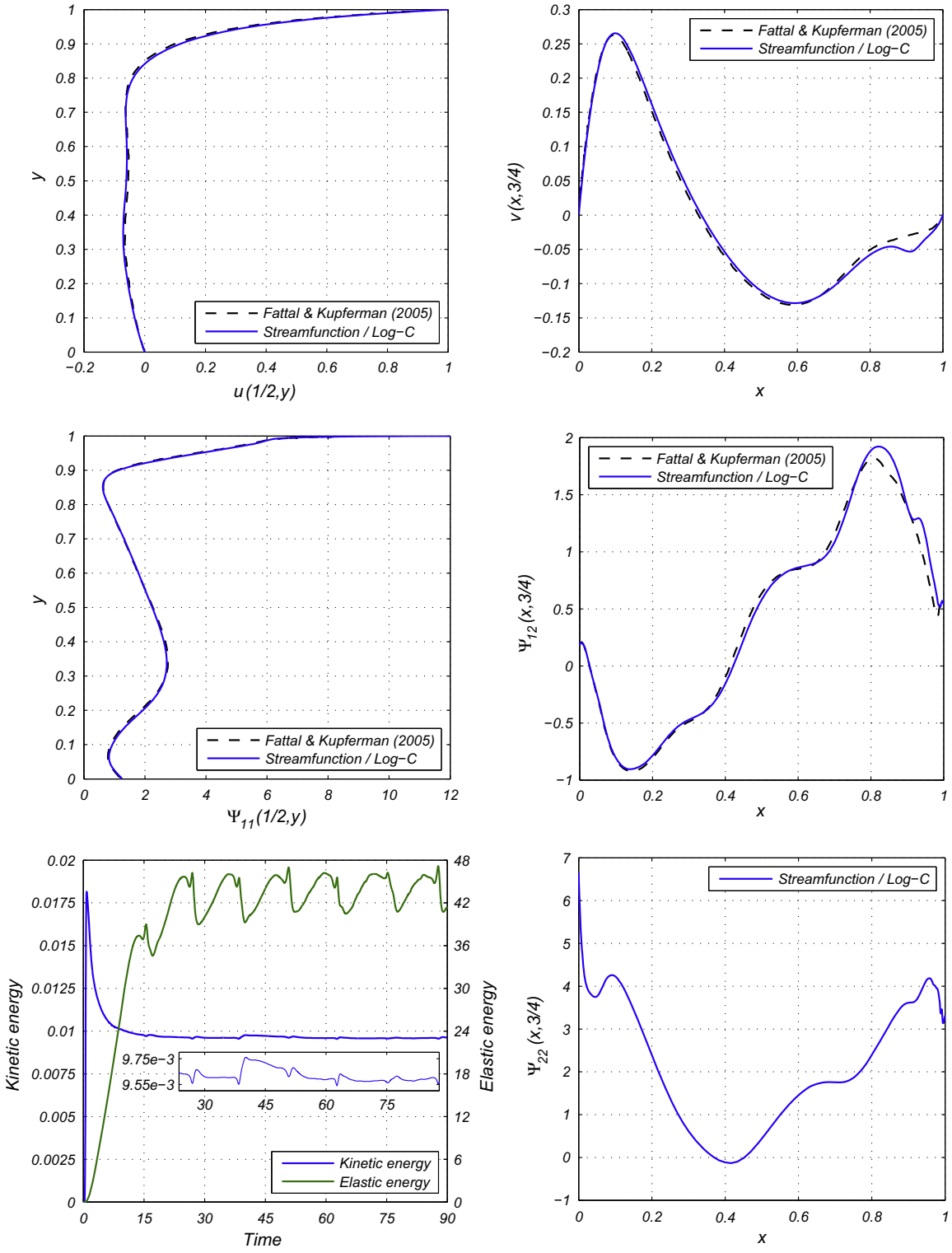


Fig. 14. Results at time $t = 40$, for $Wi = 3$: velocity and stress profiles along the lines $x = 1/2$ and $y = 3/4$ (compared with reference [36]); history of the kinetic energy and elastic energy.

numbers below 8. Above this value, the convergence of the log-conformation tensor is degraded to first order, because the time increments become too large to catch the details of the transient flow and the ramping of the lid velocity with the hyperbolic tangent function. The temporal convergence rates, for Courant numbers $C < 8$, are provided in Table 4.

It is interesting to note that stable solutions can be obtained with Courant numbers as large as $C = 64$, in contrast to the decoupled velocity–pressure methods [26]. This result demonstrates the robustness of the fully implicit streamfunction method for solving the conservation laws. At large Courant numbers, each time-step requires more iterations in order to converge, but the simulation comprises less time-steps. As we use a direct matrix solver, the computation cost is directly proportional to the total number of iterations within the entire simulation. Fig. 7 (left) compares the recorded computation costs for the different Courant numbers. It clearly shows that the use of large time-step sizes is beneficial, as far as the computation cost is concerned. However, the precision of the calculations should also be considered for assessing the computational efficiency. The absolute errors of each variable are evaluated at all the grid points, and normalized by the largest values:

$$\widehat{e}_a(\chi_C, x_i, y_j) = \frac{|\chi_C(x_i, y_j) - \tilde{\chi}_0(x_i, y_j)|}{\max_{ij}[\tilde{\chi}_0(x_i, y_j)]}, \tag{99}$$

where $\widehat{e}_a(\chi_C, x_i, y_j)$ are the normalized absolute errors, $\chi_C(x_i, y_j)$ is the numerical solution obtained with the Courant number C, and $\tilde{\chi}_0$ is the reference solution obtained with the finest time increment ($C = 1/8$). Fig. 7(right) provides the maximum values and the root-mean-squares of the normalized absolute errors of each variable. These data can be used to estimate the maximum allowable time-step size (i.e. the Courant number), in order to reach a given numerical precision.

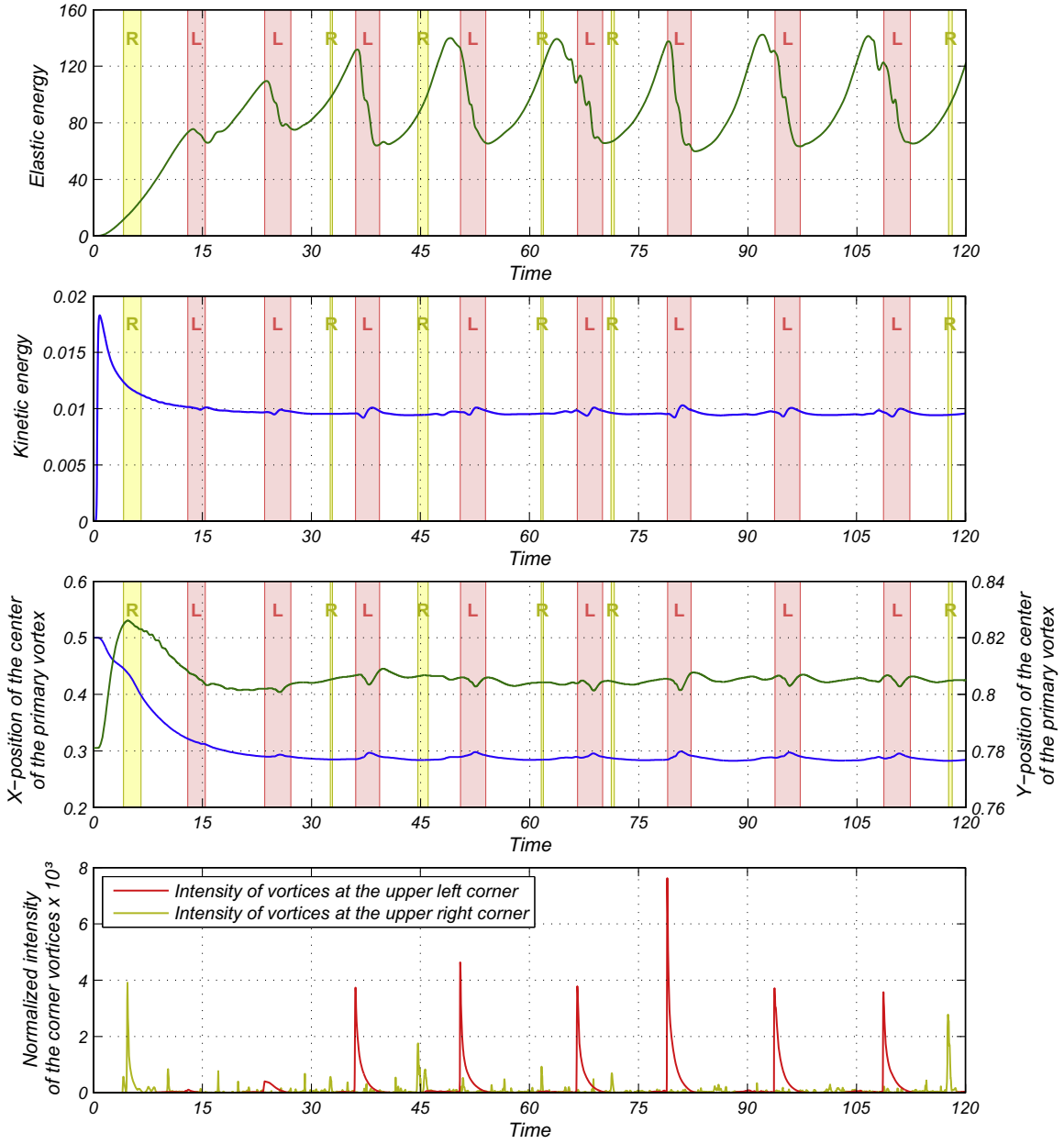


Fig. 15. History of the elastic and kinetic energy, evolution of the center of the primary vortex, and intensity of secondary vortices generated at the upper corners, for $Wi = 5$. The red and yellow colored areas correspond to the time-intervals of noticeable secondary vortices, at the upper left and right corners, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6.2. Simulations at moderate Weissenberg numbers

In this subsection, the transient viscoelastic lid-driven cavity flow is solved for the moderate Weissenberg numbers $Wi = 1$, 2 and 3. Our numerical solutions are compared with the available results in the literature [36,74]. A non-uniform Cartesian grid comprising 250×250 control volumes is used. The grid is symmetrically refined near the walls, with a constant contraction ratio $\Delta x_{k+1}/\Delta x_k = 0.96$ when approaching the walls, see Fig. 8. The maximum and minimum grid spacings are 0.02 at the center of the cavity, and 1.27×10^{-4} at the walls, respectively. The computational efficiency is enhanced with an adaptive time-stepping procedure based on the convergence rate of the residuals; see details in Appendix.

The solution of the streamlines and the components of the log-conformation tensor are represented in Figs. 9, 11 and 13, at $t = 8$ for $Wi = 1$, and $t = 40$ for $Wi = 2$ and 3, respectively. Figs. 10, 12 and 14 provide velocity and log-conformation profiles along the vertical line $x = 1/2$ and horizontal line $y = 3/4$, for $Wi = 1$, 2 and 3, respectively. These figures also display the history of the specific kinetic energy E_k, and the specific elastic energy E_e (i.e. the Helmholtz free energy), defined as:

$$E_k = \frac{1}{2}\int_\Omega (u^2 + v^2)dV, \tag{100}$$

$$E_e = \frac{1}{2}\int_\Omega \mathrm{tr}(\boldsymbol{c} - \boldsymbol{I})dV. \tag{101}$$

For $Wi = 1$ and 2, our results are in good agreements with those in [36,74]. For $Wi = 3$, we observe some disparities of the results in the vicinity of the upper right corner. Nonetheless, our solution is

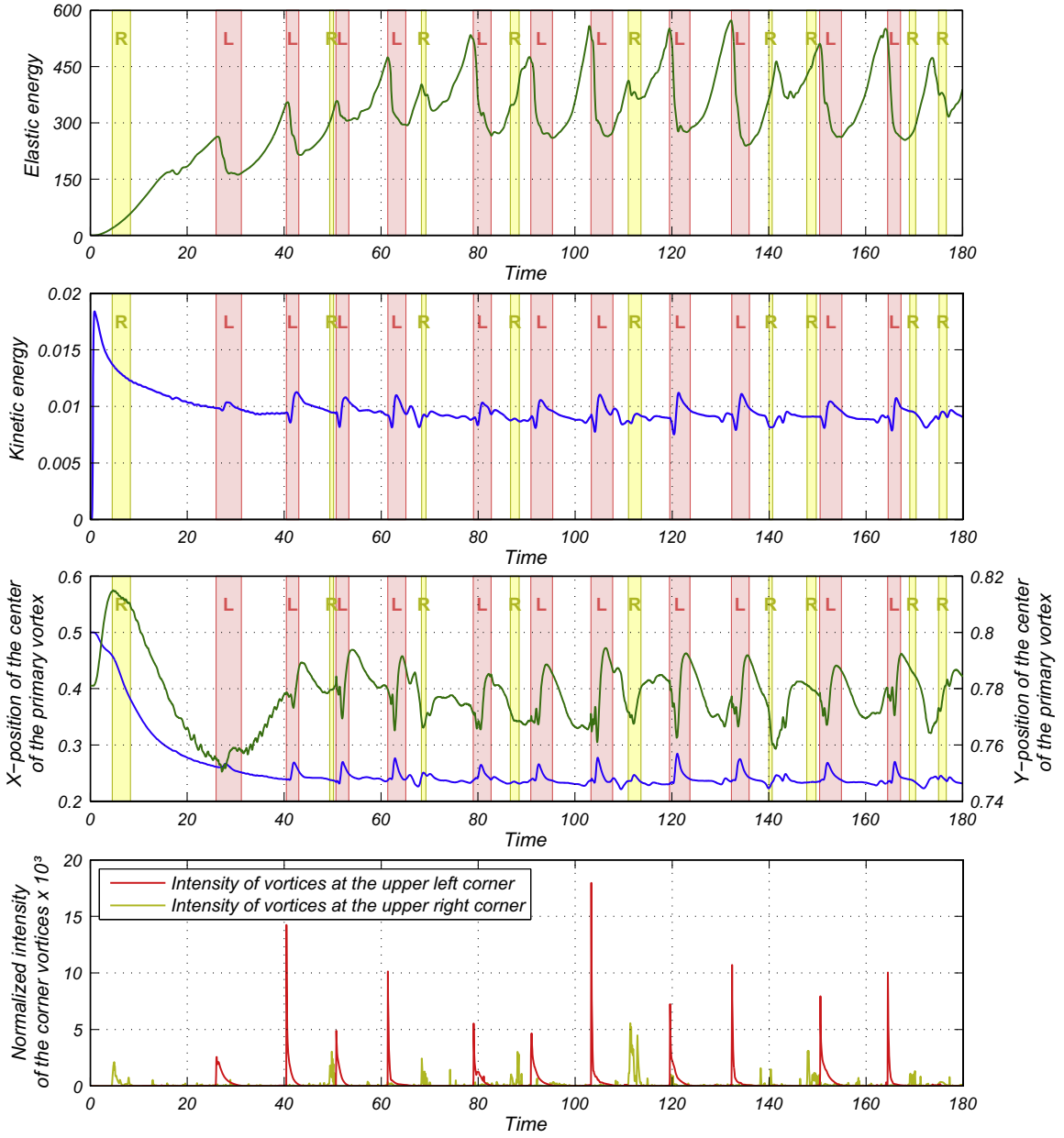


Fig. 16. History of the elastic and kinetic energy, evolution of the center of the primary vortex, and intensity of secondary vortices generated at the upper corners, for $Wi = 10$. The red and yellow colored areas correspond to the time-intervals of noticeable secondary vortices, at the upper left and right corners, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

obtained with grid spacings near the walls 30 time finer than in [36], and a second-order accurate scheme. As observed in [36], the solution for $Wi = 3$ exhibits small oscillations due to vortices repeatedly created and annihilated, in the vicinity of the upper right corner. Moreover, Fattal and Kupferman [36] reported convergence difficulties attributed to the under-resolution of subgrid vortices. Under-resolution is generally manifested by chaotic oscillations in the history of the kinetic energy or the elastic energy. However, in our simulation the history of the kinetic and elastic energies exhibits a quasi-periodicity, for $Wi \geqslant 2$. This quasi-periodicity underpins that the oscillations in the flow are likely to be linked to a structural pattern of elastic instabilities rather than numerical errors. A further discussion on the elastic instabilities is given in the next subsection.

6.3. Simulations at high Weissenberg numbers

In this subsection, the viscoelastic flow in the lid-driven cavity is solved for the high Weissenberg numbers $Wi = 5$ and 10. We use the same mesh and the same adaptive time-stepping procedure as in the previous simulations for lower Weissenberg numbers. The results for $Wi = 5$ and 10 present similar features that are described below.

After the lid velocity has reached its maximum value, the flows undergo transient developments, where the kinetic energy slowly decreases, while elastic energy gets stored in the fluid. The transient developments of the flows take longer times at high Weissenberg numbers, because the fluid has larger relaxation times. During the flow development, some vortices appear at the

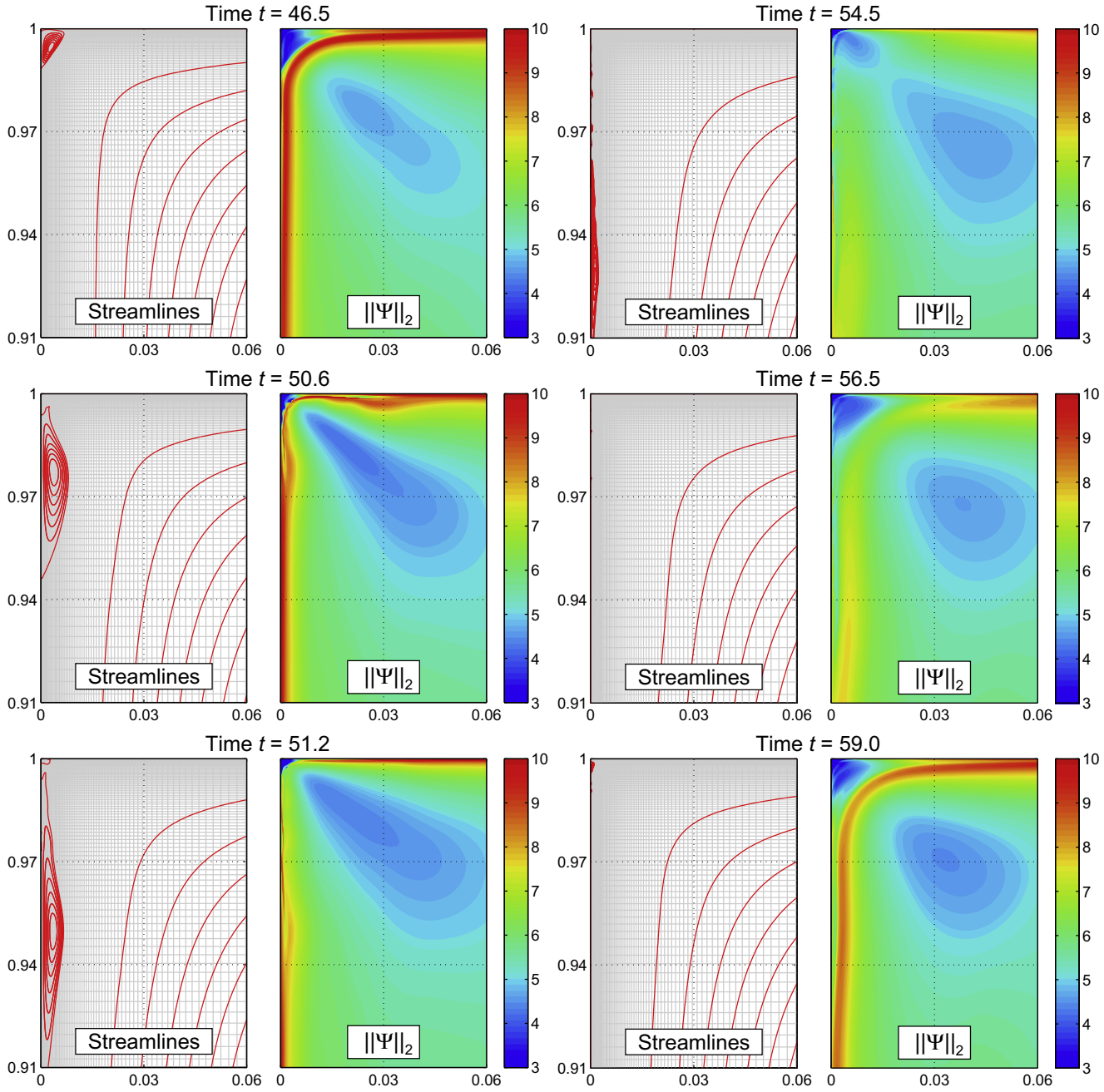


Fig. 17. Snapshots of the streamlines and the 2-norm of the log-conformation tensor at the upper left corner, for $Wi = 5$.

upper right corner and move downward along the wall, until they either annihilate or merge with the vortex at the lower corner.

At some point, the flows evolve into quasi-periodic unsteady states. The quasi-periodicity of the flows are clearly visible in the histories of the specific elastic and kinetic energy, represented in Figs. 15 and 16 for $Wi = 5$ and 10, respectively. The amplitude of the oscillations increases with the Weissenberg number, but not the quasi-periodicity, interestingly. Two videos showing the patterns of the vorticity and streamlines, for $Wi = 5$ and 10, are provided as supplementary material for this paper. Figs. 15 and 16 also display the evolution of the center of the primary vortex, as well as the normalized intensity of vortices generated at the upper corners (scaled by the intensity of the primary vortex). The red and yellow colored areas on the graphs correspond to the time-intervals of noticeable secondary vortices at the upper left and right corners, respectively. Our results show a clear correlation between the appearance of secondary vortices at the upper left corner and the flows' quasi-periodicity.

When examining the region of the upper left corner in details, we can see the presence of a small corner vortex, see Figs. 17 and 18 for $Wi = 5$ and 10, respectively. However, the corner vortex has such a low intensity that it can be consisted as virtually stagnating, within the time scale of the material relaxation. Nevertheless, the fluid at the interface between the corner vortex and the primary vortex is subjected to shear deformations. Thus, an elastic stress layer is built-up at the interface between the vortices, as it can be seen in Figs. 17 and 18, where the stress level is monitored by the entrywise 2-norm of the log-conformation tensor $||\Psi||_2 = \sqrt{\sum_{ij} \Psi_{ij}^2}$. The large elastic stress stored in this stress

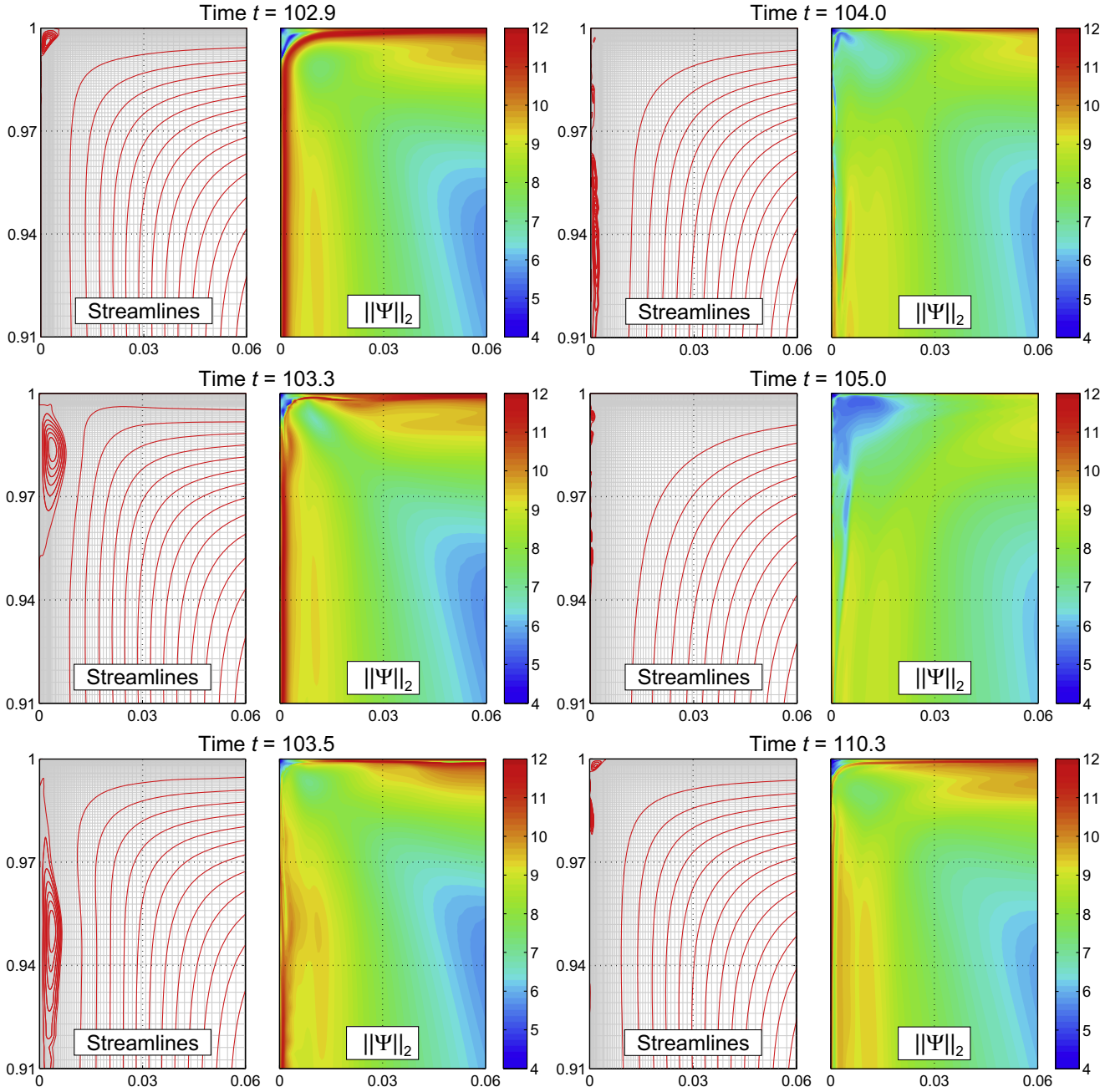


Fig. 18. Snapshots of the streamlines and the 2-norm of the log-conformation tensor at the upper left corner, for $Wi = 10$.

layer is responsible for the increases of elastic energy observed in Figs. 15 and 16. When the accumulated elastic energy reaches a threshold value, another secondary vortex appears just below the corner vortex. The secondary vortex destroys the elastic stress layer while it grows and propagates downward the wall. The elastic energy stored in the stress layer is partly converted into kinetic energy and partly dissipated through viscous dissipation. The secondary vortex also affects the primary vortex of the cavity. Its center gets pushed to the right and starts to oscillate, which results in the fluctuation in the kinetic and elastic energy, see Fig. 15 and 16. Finally, the secondary vortex annihilates when most of the energy elastic stored in the corner region is cleared, and the system finds itself at the beginning of another cycle, with the formation of a new corner vortex. To the knowledge of the authors, this structural mechanism of the elastic instabilities in upper left corner of the lid-driven cavity has never been identified in numerical simulations before. It is interesting to note that the elastic instabilities are solely initiated at the corner, where the flow has curved

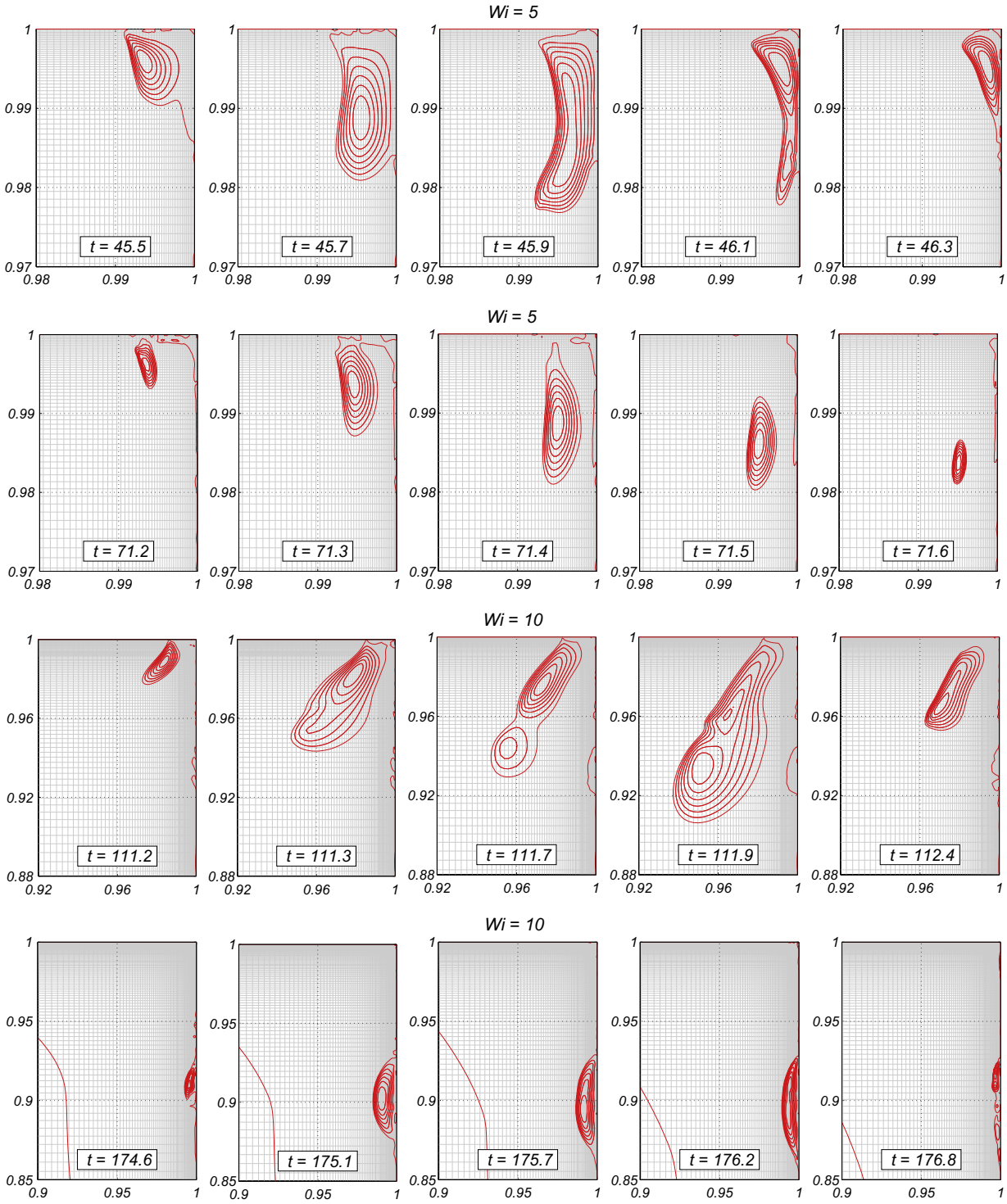


Fig. 19. Specimens of secondary vortex generated at the upper right corner, for $Wi = 5$ and 10.

streamlines, while the elastic stress layer is also built-up along the left wall and the moving lid. This feature is consistent with the general mechanism of elastic instability in curved streamlines, discussed by Pakdel and McKinley [80].

Small eddies are also generated and annihilated at the upper right corner of the cavity, see Fig. 19. However, only few of them develop a reasonable size, and they globally have no noticeable impact on the primary flow. Moreover, they do not present any regular pattern, and their occurrence seems independent from the quasi-periodic oscillations of the rest of the flow. Aperiodic fluctuations of the viscoelastic flow at the upper right corner of the lid-driven cavity have been observed experientially in [81]. Whether these fluctuations originate from elastic turbulences is still an open question.

7. Conclusions

We have presented a new formulation for incompressible viscoelastic flows, based on the streamfunction and the log-conformation as primary variables. First, the log-conformation representation obviates the high Weissenberg number problem, and enhances the stability and accuracy of the viscoelastic stress calculation. In essence, it provides a better interpolation of exponential stress profiles, and guaranties the positive-definiteness of the conformation tensor. Secondly, the streamfunction formulation provides a fully implicit pressureless method that remedies the issues related to the coupling of the velocity with pressure field. The convergence of the solution algorithm is improved and the mass conservation is automatically fulfilled to the machine precision. Furthermore, it eliminates the decoupling errors that are inherent to velocity–pressure fractional step methods, and which become large in the creeping flow regime. Therefore, numerical schemes based on the streamfunction formulation are generally more robust and accurate than those based on the classical velocity–pressure decoupled methods.

Our implementation of the streamfunction/log-conformation method with finite-volumes, quadratic interpolations and discrete curl operators, has demonstrated second-order spatial and temporal accuracy. Moreover, the fully implicit discretization and the absence of velocity–pressure decoupling errors remove the restrictions on the time-step size. Our numerical experiments show that stable calculations are attainable with Courant numbers as large as 64. Thus, the streamfunction/log-conformation method is also a progress toward more efficient and reliable viscoelastic flow solvers.

Our scheme has been used to solve the two-dimensional viscoelastic flows in the lid-driven cavity with a regularized lid velocity. The numerical results are in good agreement with the available data from the literature, for Weissenberg numbers 1, 2 and 3. However, some differences are observed in the region of the upper right corner, which we attribute to under-resolution in the previous studies, since we have a much finer mesh resolution at the corners. The viscoelastic lid-driven cavity flow has further been solved at higher Weissenberg numbers 5 and 10. Our numerical solutions present quasi-periodic oscillations due to cyclic elastic instabilities that arise at the upper left corner of the cavity. In this region, the fluid accumulates a large amount of elastic energy in a thin stress layer. Our simulations revealed the structural mechanism that sustains the quasi-periodic elastic instability by the conversion of the stored elastic energy into kinetic energy. The elastic instability arises at the corner where the streamlines have a large curvature.

In summary, we have shown that stable calculations of viscoelastic flows at high Weissenberg numbers are feasible, and can be used to simulate elastic instabilities, provided that the mesh resolution is a fine enough. We would like to outline that the str eamfunction/log-conformation framework is not tied to finite-difference or finite-volume discretizations. The streamfunc tion/log-conformation methodology could be implemented using the finite-element method, as well as higher-order discretization schemes. Finally, in addition to accuracy and robustness, the stre amfunction/log-conformation scheme is also expected to have better convergence in multi-physics simulations, since one iteration or decoupling is removed (between the velocity and the pressure). This will be beneficial when the simulation of the viscoelastic flow is coupled for instance with a thermal solver or an interface tracking procedure. There are a wide range of applications for multi-physics simulations encompassing viscoelastic flows in the fields of polymer processing and biomechanics.

Acknowledgements

The authors would like to acknowledge the support of the Scientific Research Council on Technology and Production Sciences (FTP) (Contract No. 09-072888, OPTIMAC), which is part of the Danish Council for Independent Research (DFF). The second author was funded by the FTP Contract No. 4005-00381, Decreasing concrete-related CO_2 emissions through optimization of rheology.

Appendix A. Adaptive time-stepping procedure

An adaptive time-stepping technique has been used in our simulation, in order to enhance computational efficiency. On the one hand, time-increments that are smaller than necessary result in an extension of the calculation time. On the other hand, excessively large time increments degrade the convergence of the iterations, and can cause convergence difficulties. The results in Section 6.1 show that large time-increments (requiring more iterations but fewer time-steps) are preferable to short time increments, provided that convergence is reachable. Convergence difficulties occur when the series of the residuals does not decrease anymore but oscillates around a critical value above the prescribe residual tolerance ε_{tol}. An empirical adaptive time-stepping procedure based on the convergence rate of the residuals has been developed, according to the following principles:

(i) The time increment must increase whenever possible to speed up the computation.
(ii) The time increment must preventively decrease whenever necessary in order to avoid convergence difficulties.
(iii) In case of convergence difficulties, the calculation for the current time-step must restart with a smaller time increment.

The convergence rate of the residual between two successive iterations is defined as the ratio $R_k = r_{k+1}/r_k$. The geometric mean of the convergence rates for all the iterations within a time-step is:

$$\bar{R} = \left(\frac{r_N}{r_1}\right)^{1/(N-1)}, \tag{A.1}$$

where N is the total number of iterations to reach convergence. An objective convergence rate R_{obj} is specified, together with the tolerance δR_{obj}. When the value of $\bar{R}$ falls outside the interval $\left[R_{\text{obj}} - \delta R_{\text{obj}}, R_{\text{obj}} + \delta R_{\text{obj}}\right]$, the time increment for the next time-step is adjusted according to the formula:

$$\Delta t^{n+1} = \left(\frac{R_{\text{obj}}}{\bar{R}}\right)^{q} \Delta t^{n}. \tag{A.2}$$

Table 5
Recommended parameters for the adaptive time-stepping procedure.

Symbols	Nomenclature	Recommended values
ε_{tol}	Residual tolerance	10^{-6}
R_{obj}	Objective convergence rate	0.5
δR_{obj}	Tolerance of the convergence rate	0.05
q	Exponent	2
f_{max}	Maximum increasing factor	2
f_{min}	Minimum decreasing factor	0.5
N_{max}	Maximum number of iterations	40
N_{min}	Minimum number of iterations	5
Δt_{max}	Maximum time-step size	0.1
Δt_{min}	Minimum time-step size	10^{-4}

The tolerance δR_{obj} avoids small oscillations of the time-step size at every iteration. The reactivity of the adaptive time-stepping is tuned with the exponent q. For instance, exponents larger than unity produce faster increases than decreases of the time increment. In addition, the variations of the time-step size are limited by specified maximum and minimum variation factors f_{min} and f_{max}. In summary the adaptive time-stepping proceeds as follows:

- if $R_{obj} + \delta R_{obj} \leqslant \bar{R} \leqslant R_{obj} + \delta R_{obj}$, the time-step size is kept constant for the next time-step,
- if $\bar{R} \leqslant R_{obj} f_{max}^{-1/q}$, the time-step size is increased as $\Delta t^{n+1} = f_{max} \Delta t^n$,
- if $R_{obj} f_{min}^{-1/q} \leqslant \bar{R}$, the time-step size is decreased as $\Delta t^{n+1} = f_{min} \Delta t^n$,
- Otherwise, the time-step size is adapted using the formula (A.2).

Furthermore, a convergence difficulty is detected when the number of iterations within the current time-step either exceeds a specified maximum number N_{max}, or f_{max} times the number of iterations of the previous time-step. In that case, the series of iterations is stopped and the all the calculations of the current time-step are restarted from the first iteration, with a smaller time-step size reduced by the factor f_{min}. If on the contrary the iteration series converges with less iterations than a specified minimum number N_{min}, then the time-step size for the next time-step is automatically increased by the factor f_{max}. Finally, the time-step size itself is bounded by a maximum and a minimum value Δt_{max} and Δt_{min}. Table 5 provides recommended parameters of the adaptive time-stepping, which are used in our simulation.

Appendix B. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jnnfm.2015.05.003.

References

[1] M.D. Graham, Drag reduction in turbulent flow of polymer solutions, Rheol. Rev. 2 (2004) 143–170.
[2] M.M. Denn, Fifty years of non-Newtonian fluid dynamics, AIChE J. 50 (2004) 2335–2345.
[3] R.G. Larson, Instabilities in viscoelastic flows, Rheol. Acta 31 (1992) 213–263.
[4] A. Groisman, V. Steinberg, Elastic turbulence in a polymer solution flow, Nature 405 (2000) 53–55.
[5] A.N. Morozov, W. van Saarloos, An introductory essay on subcritical instabilities and the transition to turbulence in visco-elastic parallel shear flows, Phys. Rep. 447 (2007) 112–143.
[6] X.-B. Li, F.-C. Li, W.-H. Cai, H.-N. Zhang, J.-C. Yang, Very-low-Re chaotic motions of viscoelastic fluid and its unique applications in microfluidic devices: a review, Exp. Therm. Fluid Sci. 39 (2012) 1–16.
[7] Y. Liang, A. Oztekin, S. Neti, Dynamics of viscoelastic jets of polymeric liquid extrudate, J. Non-Newton. Fluid Mech. 81 (1999) 105–132.
[8] R.I. Tanner, A theory of die-swell revisited, J. Non-Newton. Fluid Mech. 129 (2005) 85–87.
[9] M.M. Denn, Issues in viscoelastic fluid mechanics, Annu. Rev. Fluid Mech. 22 (1990) 13–32.
[10] B. Meulenbroek, C. Storm, V. Bertola, C. Wagner, D. Bonn, W. van Saarloos, Intrinsic route to melt fracture in polymer extrusion: a weakly nonlinear subcritical instability of viscoelastic Poiseuille flow, Phys. Rev. Lett. 90 (2003) 024502.
[11] E.J. Hinch, O.J. Harris, J.M. Rallison, The instability mechanism for two elastic liquids being co-extruded, J. Non-Newton. Fluid Mech. 43 (1992) 311–324.
[12] H.K. Ganpule, B. Khomami, An investigation of interfacial instabilities in the superposed channel flow of viscoelastic fluids, J. Non-Newton. Fluid Mech. 81 (1999) 27–69.
[13] P.D. Anderson, J. Dooley, H.E. Meijer, Viscoelastic effects in multilayer polymer extrusion, Appl. Rheol. 16 (2006) 198–205.
[14] P. Yue, C. Zhou, J. Dooley, J.J. Feng, Elastic encapsulation in bicomponent stratified flow of viscoelastic fluids, J. Rheol. 52 (2008) 1027–1042.
[15] J.M. Dealy, Weissenberg and Deborah numbers—their definition and use, Rheol. Bull. 79 (2010) 14–18.
[16] H. Watanabe, Viscoelasticity and dynamics of entangled polymers, Prog. Polym. Sci. 24 (1999) 1253–1403.
[17] T.C.B. McLeish, Tube theory of entangled polymer dynamics, Adv. Phys. 51 (2002) 1379–1527.
[18] R. Keunings, Micro-macro methods for the multiscale simulation of viscoelastic flow using molecular models of kinetic theory, Rheol. Rev. 2 (2004) 67–98.
[19] R.G. Larson, A critical comparison of constitutive equations for polymer melts, J. Non-Newton. Fluid Mech. 23 (1987) 249–269.
[20] J.M. Rallison, E.J. Hinch, Do we understand the physics in the constitutive equation?, J. Non-Newton. Fluid Mech. 29 (1988) 37–55.
[21] A.I. Leonov, Analysis of simple constitutive equations for viscoelastic liquids, J. Non-Newton. Fluid Mech. 42 (1992) 323–350.
[22] Y. Kwon, A.I. Leonov, Stability constraints in the formulation of viscoelastic constitutive equations, J. Non-Newton. Fluid Mech. 58 (1995) 25–46.
[23] R. Keunings, On the high Weissenberg number problem, J. Non-Newton. Fluid Mech. 20 (1986) 209–226.
[24] R. Keunings, A survey of computational rheology, in: D.M. Binding et al. (Eds.), Proceedings of the XIIIth International Congress on Rheology, vol. 1, British Soc. Rheol., Cambridge, 2000, pp. 7–14.
[25] K. Walters, M.F. Webster, The distinctive CFD challenges of computational rheology, Int. J. Numer. Meth. Fluids 43 (2003) 577–596.
[26] S.C. Xue, R.I. Tanner, N. Phan-Thien, Numerical modelling of transient viscoelastic flows, J. Non-Newton. Fluid Mech. 123 (2004) 33–58.
[27] R. Kupferman, On the linear stability of plane Couette flow for an Oldroyd-B fluid and its numerical approximation, J. Non-Newton. Fluid Mech. 127 (2005) 169–190.
[28] A.W. El-Kareh, L.G. Leal, Existence of solutions for all Deborah numbers for a non-Newtonian model modified to include diffusion, J. Non-Newton. Fluid Mech. 33 (1989) 257–287.
[29] T. Min, J.Y. Yoo, H. Choi, Effect of spatial discretization schemes on numerical solutions of viscoelastic fluid flows, J. Non-Newton. Fluid Mech. 100 (2001) 27–47.
[30] K.D. Housiadas, Stress diffusion and high order viscoelastic effects in the 3D flow past a sedimenting sphere subject to orthogonal shear, Rheol. Acta 53 (2014) 537–548.
[31] R. Sureshkumar, A.N. Beris, Effect of artificial stress diffusivity on the stability of numerical calculations and the flow dynamics of time-dependent viscoelastic flows, J. Non-Newton. Fluid Mech. 60 (1995) 53–80.
[32] B. Thomases, An analysis of the effect of stress diffusion on the dynamics of creeping viscoelastic flow, J. Non-Newton. Fluid Mech. 166 (2011) 1221–1228.
[33] M.A. Hulsen, A.P.G. van Heel, B.H.A.A. van den Brule, Simulation of viscoelastic flows using Brownian configuration fields, J. Non-Newton. Fluid Mech. 70 (1997) 79–101.
[34] J. Lee, S. Yoon, Y. Kwon, S. Kim, Practical comparison of differential viscoelastic constitutive equations in finite element analysis of planar 4:1 contraction flow, Rheol. Acta 44 (2004) 188–197.
[35] R. Fattal, R. Kupferman, Constitutive laws for the matrix-logarithm of the conformation tensor, J. Non-Newton. Fluid Mech. 123 (2004) 281–285.
[36] R. Fattal, R. Kupferman, Time-dependent simulation of viscoelastic flows at high Weissenberg number using the log-conformation representation, J. Non-Newton. Fluid Mech. 126 (2005) 23–37.
[37] M.A. Hulsen, R. Fattal, R. Kupferman, Flow of viscoelastic fluids past a cylinder at high Weissenberg number: stabilized simulations using matrix logarithms, J. Non-Newton. Fluid Mech. 127 (2005) 27–39.
[38] M. Renardy, Current issues in non-Newtonian flows: a mathematical perspective, J. Non-Newton. Fluid Mech. 90 (2000) 243–259.
[39] A. Afonso, P.J. Oliveira, F.T. Pinho, M.A. Alves, The log-conformation tensor approach in the finite-volume method framework, J. Non-Newton. Fluid Mech. 157 (2009) 55–65.
[40] M. Sahin, A stable unstructured finite volume method for parallel large-scale viscoelastic fluid flow calculations, J. Non-Newton. Fluid Mech. 166 (2011) 779–791.

[41] M.F. Tomé, A. Castelo, A.M. Afonso, M.A. Alves, F.T. Pinho, Application of the log-conformation tensor to three-dimensional time-dependent free surface flows, J. Non-Newton. Fluid Mech. 175 (2012) 44–54.
[42] F. Habla, M.W. Tan, J. Haßlberger, O. Hinrichsen, Numerical simulation of the viscoelastic flow in a three-dimensional lid-driven cavity using the log-conformation reformulation in OpenFOAM®, J. Non-Newton. Fluid Mech. 212 (2014) 47–62.
[43] Y. Kwon, Finite element analysis of planar 4:1 contraction flow with the tensor-logarithmic formulation of differential constitutive equations, Korea–Aust. Rheol. J. 16 (2004) 183–191.
[44] R. Guénette, A. Fortin, A. Kane, J.F. Hétu, An adaptive remeshing strategy for viscoelastic fluid flow simulations, J. Non-Newton. Fluid Mech. 153 (2008) 34–45.
[45] T.W. Pan, J. Hao, R. Glowinski, On the simulation of a time-dependent cavity flow of an Oldroyd-B fluid, Int. J. Numer. Meth. Fluids 60 (2009) 791–808.
[46] H. Damanik, J. Hron, A. Ouazzi, S. Turek, A monolithic FEM approach for the log-conformation reformulation (LCR) of viscoelastic flow problems, J. Non-Newton. Fluid Mech. 165 (2010) 1105–1113.
[47] L. Carracciuolo, D. Casaburi, L. D'Amore, G. D'Avino, P.L. Maffettone, A. Murli, Computational simulations of 3D large-scale time-dependent viscoelastic flows in high performance computing environment, J. Non-Newton. Fluid Mech. 166 (2011) 1382–1395.
[48] P. Knechtges, M. Behr, S. Elgeti, Fully-implicit log-conformation formulation of constitutive laws, J. Non-Newton. Fluid Mech. 214 (2014) 78–87.
[49] T. Vaithianathan, L.R. Collins, Numerical approach to simulating turbulent flow of a viscoelastic polymer solution, J. Comput. Phys. 187 (2003) 1–21.
[50] A. Lozinski, R.G. Owens, An energy estimate for the Oldroyd B model: theory and applications, J. Non-Newton. Fluid Mech. 112 (2003) 161–176.
[51] N. Balci, B. Thomases, M. Renardy, C.R. Doering, Symmetric factorization of the conformation tensor in viscoelastic fluid models, J. Non-Newton. Fluid Mech. 166 (2011) 546–553.
[52] A.M. Afonso, F.T. Pinho, M.A. Alves, The kernel-conformation constitutive laws, J. Non-Newton. Fluid Mech. 167 (2012) 30–37.
[53] X. Chen, H. Marschall, M. Schäfer, D. Bothe, A comparison of stabilisation approaches for finite-volume simulation of viscoelastic fluid flow, Int. J. Comput. Fluid D. 27 (2013) 229–250.
[54] R. Kupferman, A central-difference scheme for a pure stream function formulation of incompressible viscous flow, SIAM J. Sci. Comput. 23 (2001) 1–18.
[55] M.H. Kobayashi, J.M.C. Pereira, A computational stream function method for two-dimensional incompressible viscous flows, Int. J. Numer. Meth. Eng. 62 (2005) 1950–1981.
[56] D. Fishelov, M. Ben-Artzi, J.P. Croisille, Recent developments in the pure streamfunction formulation of the Navier–Stokes system, J. Sci. Comput. 45 (2010) 238–258.
[57] Z.F. Tian, P.X. Yu, An efficient compact difference scheme for solving the streamfunction formulation of the incompressible Navier–Stokes equations, J. Comput. Phys. 230 (2011) 6404–6419.
[58] M.D. Chilcott, J.M. Rallison, Creeping flow of dilute polymer solutions past cylinders and spheres, J. Non-Newton. Fluid Mech. 29 (1988) 381–432.
[59] J.R. Clermont, Analysis of incompressible three-dimensional flows using the concept of stream tubes in relation with a transformation of the physical domain, Rheol. Acta 27 (1988) 357–362.
[60] J.R. Clermont, M. Normandin, Numerical simulation of extrudate swell for Oldroyd-B fluids using the stream-tube analysis and a streamline approximation, J. Non-Newton. Fluid Mech. 50 (1993) 193–215.
[61] S. Wang, G. He, X. Zhang, Parallel computing strategy for a flow solver based on immersed boundary method and discrete stream-function formulation, Comput. Fluids 88 (2013) 210–224.
[62] R.B. Bird, J.M. Wiest, Constitutive equations for polymeric liquids, Annu. Rev. Fluid Mech. 27 (1995) 169–193.
[63] F. Dupret, J.M. Marchal, Loss of evolution in the flow of viscoelastic fluids, J. Non-Newton. Fluid Mech. 20 (1986) 143–171.
[64] A. Arsac, C. Carrot, J. Guillet, P. Revenu, Problems originating from the use of the Gordon–Schowalter derivative in the Johnson–Segalman and related models in various shear flow situations, J. Non-Newton. Fluid Mech. 55 (1994) 21–36.
[65] H.S. da Costa Mattos, Some remarks about the use of Gordon–Schowalter time derivative in rate-type viscoelastic constitutive equations, Int. J. Eng. Sci. 73 (2013) 56–66.
[66] P.C. Bollada, T.N. Phillips, On the mathematical modelling of a compressible viscoelastic fluid, Arch. Ration. Mech. An. 205 (2012) 1–26.
[67] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys. 108 (1993) 51–58.
[68] P.J. Oliveira, A traceless stress tensor formulation for viscoelastic fluid flow, J. Non-Newton. Fluid Mech. 95 (2000) 55–65.
[69] Y. Kwon, Numerical description of elastic flow instability and its dependence on liquid viscoelasticity in planar contraction, J. Rheol. 56 (2012) 1335–1362.
[70] M.J. Ni, M.A. Abdou, A bridge between projection methods and SIMPLE type methods for incompressible Navier–Stokes equations, Int. J. Numer. Meth. Eng. 72 (2007) 1490–1512.
[71] W. Chang, F. Giraldo, B. Perot, Analysis of an exact fractional step method, J. Comput. Phys. 180 (2002) 183–199.
[72] M.A. Alves, P.J. Oliveira, F.T. Pinho, A convergent and universally bounded interpolation scheme for the treatment of advection, Int. J. Numer. Meth. Fluids 41 (2003) 47–75.
[73] K. Kim, S.J. Baek, H.J. Sung, An implicit velocity decoupling procedure for the incompressible Navier–Stokes equations, Int. J. Numer. Meth. Fluids 38 (2002) 125–138.
[74] J. Hao, T.W. Pan, Simulation for high Weissenberg number: viscoelastic flow by a finite element method, Appl. Math. Lett. 20 (2007) 988–993.
[75] F. Belblidia, H. Matallah, B. Puangkird, M.F. Webster, Alternative subcell discretisations for viscoelastic flow: stress interpolation, J. Non-Newton. Fluid Mech. 146 (2007) 59–78.
[76] M. Renardy, Stress integration for the constitutive law of the upper convected Maxwell fluid near the corners in a driven cavity, J. Non-Newton. Fluid Mech. 112 (2003) 77–84.
[77] M. Renardy, A comment on smoothness of viscoelastic stresses, J. Non-Newton. Fluid Mech. 138 (2006) 204–205.
[78] J. Koplik, J.R. Banavar, Corner flow in the sliding plate problem, Phys. Fluids 7 (1995) 3118–3125.
[79] R.L. Thompson, P.R.S. Mendes, Persistence of straining and flow classification, Int. J. Eng. Sci. 43 (2005) 79–105.
[80] P. Pakdel, G.H. McKinley, Elastic instability and curved streamlines, Phys. Rev. Lett. 77 (1996) 2459–2462.
[81] P. Pakdel, G.H. McKinley, Cavity flows of elastic liquids: purely elastic instabilities, Phys. Fluids 10 (1998) 1058–1070.